**OTOMATISASI PERINGKASAN TEKS PADA DOKUMEN HUKUM MENGGUNAKAN METODE *LATENT SEMANTIC ANALYSIS***

SKRIPSI

Oleh:

MILLENIA RUSBANDI NIM. 1641720029



PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

**2020**

HALAMAN PENGESAHAN

**OTOMATISASI PERINGKASAN TEKS PADA DOKUMEN HUKUM MENGGUNAKAN METODE *LATENT SEMANTIC ANALYSIS***

Disusun oleh:

MILLENIA RUSBANDI NIM. 1641720029

**Proposal Skripsi ini telah diuji pada ………….**

**Disetujui oleh:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | Penguji I | : | Pramana Yoga S, S.Kom., M.MT.  NIP. 19880504 201504 1 004 | ......................... |
|  |  |  |  |  |
| 2. | Penguji II | : | Dika Rizky Yunianto, S.Kom., M.Kom.  NIP. 19920606 201903 1 017 | ......................... |
|  |  |  |  |  |
| 3. | Pembimbing I | : | Imam Fahrur Rozi, S.T., M.T.  NIP. 19840610 200812 1 004 | ......................... |
| 4. | Pembimbing II | : | Kadek Suarjuna Batubulan, S.Kom., M.T.  NIP. 19900320 201903 2 011 | ......................... |

Mengetahui,

|  |  |
| --- | --- |
| Ketua Jurusan  Teknologi Informasi | Ketua Program Studi  Teknik Informatika |
| Rudy Ariyanto, S.T., M.CS  NIP. 19711110 199903 1 002 | Imam Fahrur Rozi, ST., MT.  NIP. 198406102008121004 |

# KATA PENGANTAR

Puji syukur Alhamdulillah penulis ucapkan kehadirat Allah SWT yang telah melimpahkan rahmat, taufik, dan hidayah-Nya sehingga laporan skripsi yang berjudul “” dapat selesai tepat waktu. Penyusunan laporan skripsi digunakan untuk memenuhi persyaratan kelulusan Program Diploma IV, Jurusan Teknologi Informasi. Program Studi Teknik Informatika, Politeknik Negeri Malang. Penulis menyampaikan terimakasih atas kontribusi dalam penyelesaian skripsi ini. Dengan kerendahan hati penulis menyampaikan terima kasih kepada:

1. Bapak selaku Direktur Politeknik Negeri Malang.
2. Bapak selaku Ketua Jurusan Teknologi Informasi Politeknik Negeri Malang.
3. Bapak selaku Ketua Program Studi Teknik Informatika Politeknik Negeri Malang.
4. Ibu selaku Dosen Pembimbing I Skripsi.
5. Bapak selaku Dosen Pembimbing II Skripsi.
6. Kedua orang tua dan keluarga
7. Rekan-rekan jurusan Teknologi Informasi khususnya Program Studi Teknologi Informasi Politeknik Negeri Malang yang telah banyak membantu.
8. Bapak yang telah membantu pada penelitian sebagai narasumber dan pengolahan data.
9. Seluruh pihak yang telah membantu dan mendukung lancarnya pembuatan laporan akhir dari awal sampai selesai.

# PERNYATAAN KEASLIAN

# DAFTAR ISI

[KATA PENGANTAR 3](#_Toc43733209)

[PERNYATAAN KEASLIAN 4](#_Toc43733210)

[DAFTAR ISI 5](#_Toc43733211)

[DAFTAR TABEL 8](#_Toc43733212)

[DAFTAR GAMBAR 10](#_Toc43733213)

[DAFTAR SOURCE CODE 11](#_Toc43733214)

[1 BAB I PENDAHULUAN 12](#_Toc43733215)

[1.1 Latar Belakang 12](#_Toc43733216)

[1.2 Rumusan Masalah 13](#_Toc43733217)

[1.3 Tujuan 13](#_Toc43733218)

[1.4 Batasan Masalah 13](#_Toc43733219)

[1.5 Manfaat 14](#_Toc43733220)

[1.6 Tata Cara Penulisan 14](#_Toc43733221)

[2 BAB II TINJAUAN PUSTAKA 16](#_Toc43733222)

[2.1 Penelitian Terdahulu 16](#_Toc43733223)

[2.2 *Text Mining* 17](#_Toc43733224)

[2.3 Peringkasan Teks Otomatis 17](#_Toc43733225)

[2.4 Pre-Processing 18](#_Toc43733226)

[2.5 TF-IDF 19](#_Toc43733227)

[2.6 *Latent Semantic Analysis* 20](#_Toc43733228)

[3 BAB III METODOLOGI PENELITIAN 21](#_Toc43733229)

[3.1 Metode Pengambilan Data 22](#_Toc43733230)

[3.1.1 Studi Literatur 22](#_Toc43733231)

[3.1.2 Observasi 22](#_Toc43733232)

[3.1.3 Wawancara 22](#_Toc43733233)

[3.2 Metode Pengembangan Perangkat Lunak 22](#_Toc43733234)

[3.3 Metode Pengolahan Data 24](#_Toc43733235)

[3.4 Metode Pengujian 32](#_Toc43733236)

[3.5 Analisa Hasil dan Kesimpulan 32](#_Toc43733237)

[4 BAB IV ANALISA DAN PERANCANGAN 33](#_Toc43733238)

[4.1 Analisis Sistem 33](#_Toc43733239)

[4.1.1 Analisis Kebutuhan Fungsional 33](#_Toc43733240)

[4.1.2 Kebutuhan *Non*-fungsional 34](#_Toc43733241)

[4.2 Perancangan Sistem 35](#_Toc43733242)

[4.2.2 *Use Case* Diagram 35](#_Toc43733243)

[4.2.3 *Use Case* Spesifikasi 36](#_Toc43733244)

[4.2.4 *Use Case* Skenario 37](#_Toc43733245)

[4.2.4 *Activity* Diagram 41](#_Toc43733246)

[4.2.5 ERD 45](#_Toc43733247)

[4.2.6 Alur Metode Latent Semantic Analysis 46](#_Toc43733248)

[4.2.7 Perancangan Antar muka (Interface) 46](#_Toc43733261)

[5 BAB V IMPLEMENTASI DAN PENGUJIAN 50](#_Toc43733262)

[5.1 Implementasi *Database* 50](#_Toc43733268)

[5.2 Implementasi *Source Code* 50](#_Toc43733269)

[5.2.1 Mengolah *Stopword* 51](#_Toc43733270)

[5.2.2 Mengolah Pengguna 52](#_Toc43733271)

[5.2.3 Mengolah Dokumen 54](#_Toc43733272)

[5.2.4 Melakukan Peringkasan Dokumen 55](#_Toc43733273)

[5.2.5 Melakukan Pengujian 62](#_Toc43733274)

[5.3 Implementasi Tampilan 63](#_Toc43733275)

[5.3.1 *Login* 63](#_Toc43733276)

[5.3.2 *Dashboard* 64](#_Toc43733277)

[5.3.3 Mengolah *stopword* 65](#_Toc43733278)

[5.3.4 Mengolah dokumen 65](#_Toc43733279)

[5.3.5 Mengolah pengguna 66](#_Toc43733280)

[5.3.6 Melakukan peringkasan 67](#_Toc43733281)

[5.3.7 Melakukan pengujian 69](#_Toc43733282)

[5.4 Pengujian Fungsinalitas Sistem 69](#_Toc43733283)

[5.5 Pengujian Akurasi Sistem 70](#_Toc43733284)

[6 BAB VI HASIL DAN PEMBAHASAN 71](#_Toc43733285)

[7 BAB VII KESIMPULAN DAN SARAN 72](#_Toc43733286)

[6.1. Kesimpulan (menjawab rumusan masalah, selanjtnya kesimpulan setelah pengejian poin poin dalam bentuk paragraf) 72](#_Toc43733287)

[6.2 Saran (kendala) 72](#_Toc43733288)

[8. DAFTAR PUSTAKA (minimal 10) 73](#_Toc43733296)

[LAMPIRAN 74](#_Toc43733297)

# DAFTAR TABEL

# No table of figures entries found.

[Tabel 3.1 Pembentukan Kalimat 25](#_Toc43733298)

[Tabel 3.2 Case Folding 26](#_Toc43733299)

[Tabel 3.3 Tokenizing 26](#_Toc43733300)

[Tabel 3.4 Stopword Removal 27](#_Toc43733301)

[Tabel 3.5 Stemming 27](#_Toc43733302)

[Tabel 3.6 Contoh Perhitungan DF 27](#_Toc43733303)

[Tabel 3.7 Contoh Perhitungan TF-IDF 28](#_Toc43733304)

[Tabel 3.8 Hasil Perhitungan Matriks ATA (X) 29](#_Toc43733305)

[Tabel 3.9 Hasil Perhitungan X-λI 30](#_Toc43733306)

[Tabel 3.10 Hasil Perhitungan det |X-λI| (Eigen Value) 30](#_Toc43733307)

[Tabel 3.11 Matriks VT 30](#_Toc43733308)

[Tabel 3.12 Matriks S 31](#_Toc43733309)

[Tabel 3.13 Hasil skor perhitungan Sk 31](#_Toc43733310)

[Tabel 4.1Kebutuhan Perangkat Keras 34](#_Toc43733311)

[Tabel 4.2 Kebutuhan Perangkat Lunak 34](#_Toc43733312)

[Tabel 4.3 Spesifikasi Use Case 36](#_Toc43733313)

[Tabel 4.4 Skenario - mengolah dokumen 37](#_Toc43733314)

[Tabel 4.5 Skenario - mengolah stopword 38](#_Toc43733315)

[Tabel 4.6 Skenario – melakukan peringkasan 39](#_Toc43733316)

[Tabel 4.7 Skenario – melakukan pengujian 40](#_Toc43733317)

[Tabel 4.8 Skenario – mengolah pengguna 40](#_Toc43733318)

[Tabel 5.1 Tabel pengujian fungsionalitas sistem 69](#_Toc43733319)

# DAFTAR GAMBAR

[Gambar 3.1 Bagan 21](file:///D:\skripsi%20jurnal\proposal%20skripsi\progress%20skripsi\MilleniaRusbandi-ProgressLaporanBab5_ERD-22Juni2020.docx#_Toc43733320)

[Gambar 3.2 Metode Waterfall 23](#_Toc43733321)

[Gambar 3.3 Contoh Data : Nota Pembelaan 24](#_Toc43733322)

[Gambar 4.1 Diagram Blok Perancangan 33](#_Toc43733323)

[Gambar 4.2 Desain Sistem 35](#_Toc43733324)

[Gambar 4.3 Use Case Diagram 36](#_Toc43733325)

[Gambar 4.4 Activity Diagram Pengolahan Dokumen 42](#_Toc43733326)

[Gambar 4.5 Activity Diagram Pengolahan Data Stopword 42](#_Toc43733327)

[Gambar 4.6 Activity Diagram Pengolahan Data Pengguna 43](#_Toc43733328)

[Gambar 4.7 Activity Diagram Peringkasan Dokumen 44](#_Toc43733329)

[Gambar 4.8 Activity Diagram Pengujian Peringkasan 44](#_Toc43733330)

[Gambar 4.9 Entity Relationship Diagram 45](#_Toc43733331)

[Gambar 4.10 Alur LSA 46](#_Toc43733332)

[Gambar 4.11 Mockup Mengolah Dokumen 47](#_Toc43733333)

[Gambar 4.12 Mockup Mengolah Stopword 47](#_Toc43733334)

[Gambar 4.13 Mockup Pilih Kompresi 48](#_Toc43733335)

[Gambar 4.14 Mockup Peringkasan Dokumen 49](#_Toc43733336)

[Gambar 5.1 Tampilan Login 64](#_Toc43733337)

[Gambar 5.2 Tampilan Dashboard 64](#_Toc43733338)

[Gambar 5.3 Tampilan Mengolah Stopword 65](#_Toc43733339)

[Gambar 5.4 Tampilan Mengolah Dokumen 66](#_Toc43733340)

[Gambar 5.5 Tampilan Mengolah Pengguna 67](#_Toc43733341)

[Gambar 5.6 Tampilan Pengunggahan Dokumen 67](#_Toc43733342)

[Gambar 5.7 Tampilan Pilih Tingkat Kompresi 68](#_Toc43733343)

[Gambar 5.8 Hasil Peringkasan Dokumen 68](#_Toc43733344)

[Gambar 5.9 Tampilan Pengujian 69](#_Toc43733345)

# DAFTAR SOURCE CODE

[Source Code 5.1 Menampilkan Stopword 48](#_Toc43728281)

[Source Code 5.2 Menambahkan Stopword 48](#_Toc43728282)

[Source Code 5.3 Mengubah Stopword 49](#_Toc43728283)

[Source Code 5.4 Menghapus Stopword 49](#_Toc43728284)

[Source Code 5.5 Melihat Data Pengguna 50](#_Toc43728285)

[Source Code 5.6 Menambah data pengguna 50](#_Toc43728286)

[Source Code 5.7 Mengubah status pengguna 50](#_Toc43728287)

[Source Code 5.8Me-reset password pengguna 51](#_Toc43728288)

[Source Code 5.9 Melihat dokumen 51](#_Toc43728289)

[Source Code 5.10 Menghapus dokumen 52](#_Toc43728290)

[Source Code 5.11 Mengunggah dokumen 53](#_Toc43728291)

[Source Code 5.12 Mengubah pdf menjadi teks yang dapat diolah. 53](#_Toc43728292)

[Source Code 5.13 Batas Pengambilan teks yang diolah. 54](#_Toc43728293)

[Source Code 5.14 Preprocessing 55](#_Toc43728294)

[Source Code 5.15 TF-IDF 56](#_Toc43728295)

[Source Code 5.16 Metode LSA 58](#_Toc43728296)

[Source Code 5.17 Menghitung precision 58](#_Toc43728297)

[Source Code 5.18 Menghitung Recall 58](#_Toc43728298)

[Source Code 5.19 Menghitung f-measure 59](#_Toc43728299)

# BAB I PENDAHULUAN

## Latar Belakang

Selama periode tahun 2015–2017, jumlah kasus kriminalitas di Indonesia cenderung fluktuatif (tidak stabil). Berdasarkan data Kepolisian Republik Indonesia (Polri), total jumlah kasus kriminalitas pada tahun 2015 sebanyak 352.936 kasus. Pada tahun 2016, kasus semakin meningkat menjadi sebanyak 357.197 kasus. Sedangkan pada tahun 2017, mengalami penurunan menjadi 336.652 kasus (Badan Pusat Statistik, 2018). Dari data tersebut, rata-rata jumlah kriminalitas periode tahun 2015-2017 sebesar 348.928 kasus. Banyaknya jumlah kriminalitas di Indonesia akan berdampak pada jumlah dokumen hukum yang akan ditangani oleh aparat penegak hukum.

Dalam pemahaman dokumen hukum, aparat penegak hukum seperti advokat, hakim, maupun jaksa harus membaca keseluruhan dokumen dimana akan memerlukan waktu yang cukup lama. Kapasitas yang dimiliki manusia untuk mengonsumsi informasi sangat terbatas. Oleh karena itu, sangatlah penting untuk menyaring informasi yang hanya berguna dari jumlah data tidak terstuktur. Sehingga salah satu solusi yang dibutuhkan yaitu dengan membuat sebuah ringkasan pada dokumen hukum dimana dokumen tersebut berbentuk pdf. Dokumen hukum yang akan diringkas berupa nota pembelaan dan bagian yang diolah yaitu pendahuluan. Dilakukannya peringkasan pada bagian tersebut dikarenakan berbentuk paragraf dan dapat diringkas. Sehingga kalimat-kalimat yang tidak diperlukan akan dihilangkan.

Ringkasan adalah teks yang dihasilkan dari sebuah kumpulan teks yang mengandung informasi penting dengan panjang teks tidak lebih dari setengah teks aslinya (Radev, 2002). Dalam dunia komputer, peringkasan juga dikenal dengan Peringkasan teks Otomatis atau ATS (*automatic text summarization*). Peringkasan teks adalah pembuatan bentuk yang lebih singkat dari suatu teks dengan memanfaatkan aplikasi yang dijalankan dan dioperasikan pada komputer (Nugraha, 2008). Dengan adanya ringkasan tersebut, dapat mempermudah aparat penegak hukum dalam memahami dokumen tanpa harus membaca keseluruhan isi dokumen.

Pada penelitian yang telah dilakukan, umumnya menggunakan objek berupa artikel dan berita. Sedangkan untuk dokumen hukum, masih belum banyak dilakukan. Oleh karena itu, dalam penelitian ini mengusulkan peringkasan teks pada dokumen hukum dengan menggunakan metode latent semantic analysis (LSA) diharapkan dapat menghasilkan ringkasan kalimat dengan baik dan sesuai dengan isi dari dokumen hukum tersebut.

## Rumusan Masalah

Berdasarkan latar belakang di atas, maka rumusan masalah yang dapat diambil adalah sebagai berikut :

1. Bagaimanakah membaca dan mengubah dokumen hukum dalam format pdf ke dalam teks yang dapat diolah oleh sistem peringkasan otomatis.
2. Bagaimanakah pengimplementasian metode LSA (*Latent Semantic Analysis*) padaperingkasan dokumen hukum.
3. Bagaimanakah pengujian yang dilakukan untuk mengukur akurasi peringkasan dokumen otomatis.

## Tujuan

Berdasarkan latar belakang di atas, maka rumusan masalah yang dapat diambil adalah sebagai berikut :

Membantu pihak aparat hukum dalam melakukan peringkasan dokumen.

Membaca dokumen hukum dalam format pdf dan mengubahnya ke dalam teks yang dapat diolah oleh sistem peringkasan otomatis.

Menerapkan metode *latent semantic analysis* pada ringkasan dokumen hukum.

Menganalisis performasi hasil ringkasan dari metode tersebut berupa akurasi berdasarkan *precision*, *recall* dan *f-measure*.

## Batasan Masalah

Agar penelitian ini dapat berjalan sesuai dengan rencana dan tujuan awal, maka penulis memberikan batasan-batasan masalah yaitu :

1. Data yang digunakan yaitu data berupa dokumen hukum pembelaan kasus pada Kantor Advokat Ahmad Riyadh Umar Balhmar, Surabaya, Jawa Timur, Indonesia.
2. Dokumen yang diringkas berformat pdf.
3. Bagian yang digunakan untuk peringkasan yaitu pada pendahuluan.
4. Peringkasan yang digunakan yaitu secara ekstraktif.
5. Sistem ini berbasis website.

## Manfaat

Manfaat yang diharapkan dalam penelitian ini yaitu hasil implementasi dapat membantu aparat hukum dalam melakukan peringkasan agar mempermudah dalam pemahaman dokumen hukum.

## Tata Cara Penulisan

Uraian dalam penulisan skripsi ini disusun dengan sistematika penulisan sebagai berikut :

BAB I : PENDAHULUAN

Bab ini berisikan uraian tentang hal-hal yang melatar belakangi masalah dilakukannya pembuatan penelitian dan yang menjadi dasar permasalahan. Terdiri atas latar belakang masalah, rumusan masalah, tujuan penelitian, metodologi penelitian, sistematika penulisan, dan penjadwalan kegiatan penelitian.

BAB II : LANDASAN TEORI

Pembahasan dalam bagian ini mengenai landasan teori yang menjadi referensi utama dalam melaksanakan penelitian.Teori-teori tersebut adalah teori mengenai “Otomatisasi Peringkasan Teks Pada Dokumen Hukum Menggunakan Metode *Latent Semantic Analysis*”.

BAB III : METODOLOGI PENELITIAN

Bagian ini berisi tahapan yang dilakukan untuk menyelesaikan masalah pada tugas akhir yang bersumber dari proses dalam perencanaan tugas akhir. Metodologi penelitian berisi uraian tentang metode pengambilan data, metode pengembangan sistem, fase-fase pengembangan sistem dan pengujian sistem.

BAB IV : ANALISIS DAN PERANCANGAN

Bab ini menjabarkan dan menguraikan tentang analisa dan perencanaan pembuatan keseluruhan aplikasi dari penelitian yang dilakukan, serta melakukan analisa hasil yang didapat. Serta lampiran user interface.

BAB V : IMPLEMENTASI

Bab ini menjelaskan tentang bagaimana sistem dibuat dan berjalan berdasarkan analisa dan perancangan yang telah dilakukan sebelumnya. Dimana sistem diharapkan mampu membuat “Otomatisasi Peringkasan Teks Pada Dokumen Hukum Menggunakan Metode *Latent Semantic Analysis*”.

BAB VI : PENGUJIAN DAN PEMBAHASAN

Bab ini berisikan tentang tampilan seperti form – form penginputan dan output dalam sistem “Otomatisasi Peringkasan Teks Pada Dokumen Hukum Menggunakan Metode Latent Semantic Analysis”. Selain itu dilakukan juga pembahasan tentang analisa hasil yang diperoleh dari sistem yang dibuat.

BAB VII : KESIMPULAN

Bab ini dibagi menjadi dua sub bab, kesimpulan yang menjawab permasalahan yang dihadaopi dan saran yang berisikan solusi alternatif untuk permasalahan yang terjadi pada laporan akhir ini.

BAB VIII : DAFTAR PUSTAKA

Berisikan catatan semua sumber referensi yang digunakan dalam penulisan laporan akhir.

# BAB II TINJAUAN PUSTAKA

## Penelitian Terdahulu

Beberapa penelitian terdahulu yang memiliki keterkaitan dengan penelitian ini, untuk menentukan posisi penelitian. Adapun penelitian terdahulu yang peneliti maksud adalah:

Berdasarkan penelitian pada tahun 2014, Agustinus Widiantoro telah membangun sebuah aplikasi peringkasan dokumen berbahasa jawa secara otomatis menggunakan metode TF-IDF (*Term Frequency-Inverse Document Frequency*). Dari penelitian tersebut, hasil yang didapatkan menunjukan bahwa tingkat keakurasian sistem mencapai 64%. Tetapi, kelemahan dari sistem ini yaitu kurangnya daftar kata umum (*stopwords*) yang digunakan sehingga penulis menyarankan untuk melakukan penambahan daftar kata umum untuk meningkatkan akurasi (Widiantoro, 2014).

Pada tahun 2015, penelitian dilakukan oleh Geetha J.K. dan Deepamala N. dengan judul “*Kannada text summarization using Latent Semantic Analysis*”. Penelitian tersebut menggunakan Bahasa Kanada. Tingkat akurasi yang didapatkan yaitu 94% dan *precision* sebesar 80% (N & K, 2015).

Pada tahun 2017, penelitian dilakukan oleh Sohini Roy Chowdhury , Kamal Sarkar dan Santanu Dam dengan judul “*An Approach to Generic Bengali Text Summarization Using Latent Semantic Analysis*”. Penelitian tersebut menggunakan aksara Bengali. Teks yang diringkas diambil 10% dari teks asli. Nilai F-Score yang didapatkan yaitu 0.324347.

Penelitian berikutnya dilakukan oleh Tinaliah dan Triana Elizabeth pada tahun 2018. Peneliti menggunakan metode *Latent Semantic Analysis* dan *Jaro-Winkler Distance* dalam mendeteksi plagiarisme dokumen. Tingkat akurasi pada penelitian tersebut menghasilkan nilai plagiat mencapai 97,14% (Tinaliah & Elizabeth, 2018).

Dari penelitian yang telah dipaparkan diatas, Metode *Latent Semantic Analysis* dan Metode TF-IDF dapat diterapkan pada proses peringkasan dokumen dengan tingkat akurasi yang cukup baik. Maka dari itu, pada penelitian ini penulis menggunakan dua metode tersebut dalam melakukan peringkasan teks. Untuk objek yang akan digunakan yaitu berupa dokumen hukum.

## *Text Mining*

*Text mining* memiliki definisi menambang data yang berupa teks dimana sumber data di dapatkan dari dokumen. Tujuan *text mining* adalah mencari kata-kata yang dapat mewakili isi dari dokumen dan dilakukannya analisa keterhubungan antar dokumen.

Jenis masukan (*input*) dari *text mining* berupa data tak terstruktur yang merupakan pembeda utama dari *data mining* dimana menggunakan data terstruktur atau basis data sebagai masukan. *Text mining* dapat dianggap sebagai proses dua tahap yang diawali dengan penerapan struktur terhadap sumber data teks dan dilanjutkan dengan ekstraksi informasi dan pengetahuan yang relevan dari data teks terstruktur ini dengan menggunakan teknik dan alat yang sama dengan *data mining*. Proses yang umum dilakukan oleh text mining, yaitu perangkuman otomatis, kategorisasi dokumen, penggugusan teks, dll (Nindito, 2016).

## Peringkasan Teks Otomatis

Peringkasan teks otomatis merupakan pembuatan rangkuman dari suatu teks secara otomatis dengan menggunakan serta memanfaatkan sistem peringkasan teks yang dijalankan pada komputer. Sebuah sistem peringkasan teks diberi *input* (masukan) berupa teks kemudian sistem akan memproses dengan melakukan peringkasan yang akan menghasilkan *output* (keluaran) berupa teks yang lebih singkat dari sumber teks aslinya (Hovy, 2005).Terdapat dua pendekatan peringkas teks yaitu:

1. Ekstraksi (*extractive summary*)

Pada teknik ekstraksi, sistem menyalin unit-unit yang dianggap paling penting dari sebuah teks dan diubah menjadi ringkasan. Unit-unit teks yang disalin dapat berupa klausa utama, kalimat utama, atau paragraf utama tanpa ada penambahan kalimat-kalimat baru yang terdapat pada dokumen aslinya.

1. Abstraksi (*abstractive summary*)

Teknik abstraksi menggunakan metode *linguistic* untuk memeriksa dan menafsirkan teks menjadi ringkasan. Ringkasan teks tersebut dihasilkan dengan cara menambahkan kalimat-kalimat baru yang merepresentasikan intisari teks sumber ke dalam bentuk yang berbeda dengan kalimat-kalimat yang ada pada teks sumber (Gupta & Lehal, 2010).

Pada penelitian ini, metode yang digunakan untuk melakukan peringkasan teks otomatis adalah dengan menggunakan teknik ekstraksi. Hasil dari ringkasan merupakan kalimat asli yang terdapat pada dokumen dan tidak mengalami penambahan kalimat.

## **Pre-Processing**

Pada tahapan ini, data tekstual akan diubah menjadi teks agar dapat diolah oleh sistem. Penelitian ini menggunakan dokumen sebagai inputan awal. Proses yang digunakan antara lain :

1. Pembentukan Kalimat

Pembentukan kalimat yaitu pemecahan teks dokumen menjadi kumpulan kalimat berdasarkan delimiter.

1. *Case Folding*

*Case* folding merupakan pengubahan huruf pada kalimat menjadi huruf kecil (*lowercase*) dan penghilangan karakter yang tidak valid seperti tanda baca.

1. *Tokenizing*

Pada proses ini, kalimat tersebut dipecah kembali menjadi beberapa kata tunggal penyusunnya.

1. *Stopword Removal*

*Stopword removal* adalah proses penghilangan kata-kata yang tidak merepresentasikan isi dokumen.

1. *Stemming*

Stemming adalah proses pengembalian kata tunggal yang memiliki imbuhan menjadi kata dasar.

## **TF-IDF**

Setelah dokumen diproses dengan cara preprocessing, tokenizing, filtering dan stemming, selanjutnya dilakukan proses pembobotan kata. Pada Metode ini pembobotan kata dalam sebuah dokumen dilakukan dengan mengalikan nilai TF dan IDF.

Term frequency (TF) adalah pengukuran yang paling sederhana dalam metode pembobotan. Pada metode ini, masing-masing term diasumsikan mempunyai proporsi kepentingan sesuai jumlah kemunculan dalam teks dokumen. Term frequency dapat memperbaiki nilai recall pada information retrieval, tetapi tidak selalu memperbaiki nilai precision (Tokunaga & Iwayama, 1994). Hal ini disebabkan term yang frequent cenderung muncul di banyak teks, sehingga term tersebut memiliki kekuatan

Inverse document frequency (IDF) adalah metode pembobotan term yang lebih condong (fokus) untuk memperhatikan kemunculan term pada keseluruhan kumpulan teks. Pada IDF, term yang jarang muncul pada keseluruhan koleksi teks dinilai lebih berharga. Nilai kepentingan tiap term diasumsikan berbanding terbalik dengan jumlah teks yang mengandung term tersebut (Tokunaga & Iwayama, 1994).

Term frequncy inverse document frequency (TF•IDF) adalah metode pembobotan yang menggabungkan metode TF dan IDF. Metode ini diusulkan oleh Salton sebagai sebuah kombinasi metode yang dapat memberikan performansi yang lebih baik, khususnya dalam memperbaiki nilai recall dan precision (Tokunaga & Iwayama, 1994). Berikut ini merupakan perhitungannya :

TF.IDF = TF \* log(N/DF) (2.1)

Keterangan :

TF : Jumlah *term* tersebut  
N : Total dokumen  
DF : Jumlah dokumen yang mengandung suatu *term*

## *Latent Semantic Analysis*

*Latent Semantic Analysis* (LSA) menurut bahasa terbagi atas beberapa kata yang penting yaitu *latent* dan *semantic*, *latent* yang memiliki arti tersembunyi atau sesuatu yang masih belum terlihat, sedangkan semantic berasal dari bahasa yunani “*semanticos*” yang berarti memberi tanda, penting atau cabang linguistik yang mempelajari arti dan makna dari suatu bahasa, kode atau jenis representasi lainnya.

Dari pengertian dapat ditarik kesimpulan bahwa, LSA adalah menguraikan atau menganalisa makna yang masih tersembunyi dari suatu bahasa, kode atau jenis representasi lainnya, guna memperoleh informasi yang penting. Kesamaan kata dan kalimat diperoleh dengan cara menggunakan *Singular Value Decomposition* (SVD), di mana SVD mempunyai kapasitas untuk mereduksi *noise*, sehingga dapat meningkatkan hasil akurasi pada ringkasan (Peter & Kp, 2009).

# BAB III METODOLOGI PENELITIAN

Dalam metodologi penelitian, menjelaskan langkah-langkah yang dilakukan untuk merancang sistem. Metodologi penelitian yang dilakukan dalam penelitian ini melalui beberapa tahap, yaitu studi literature, pengumpulan data, analisa dan perancangan, implementasi sistem, uji coba sistem, dan kesimpulan. Tahapan dalam penelitian ini terdapat pada bagan berikut.

Gambar 3.1 Bagan

Studi Literatur

Pengumpulan Data

Analisa dan Perancangan

Implementasi

Pengujian dan Analisis

Pengambilan Kesimpulan

## Metode Pengambilan Data

Metode pengambilan data digunakan untuk mengumpulkan data-data yang dibutuhkan dalam proses penelitian. Pemgambilan data dengan cara antara lain studi literatur.

### Studi Literatur

Studi literatur digunakan untuk mencari informasi pustaka yang berkaitan dengan penelitian ini, yang diperoleh dari jurnal maupun penelitian sebelumnya meliputi metode – metode yang dilakukan dalam melakukan peringkasan dokumen.

### Observasi

Dalam pengambilan data, Penulis langsung terjun ke lapangan Kantor Advokat Ahmad Riyadh Umar Balhmar Kota Surabaya untuk melakukan observasi.

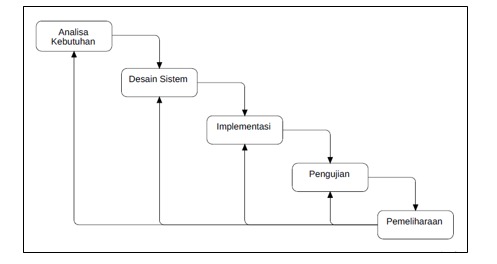
### Wawancara

Penulis melakukan kegiatan tanya jawab dengan pihak Advokat Kota Surabaya.

## Metode Pengembangan Perangkat Lunak

Pengembangan perangkat lunak dapat diartikan sebagai proses membuat suatu perangkat lunak baru untuk menggantikan perangkat lunak lama secara keseluruhan atau memperbaiki perangkat lunak yang telah ada. Agar lebih cepat dan tepat dalam mendeskripsikan solusi dan mengembangkan perangkat lunak   juga hasilnya mudah dikembangkan dan dipelihara, maka pengembangan perangkat lunak memerlukan suatu metodologi khusus. Metodologi pengembangan perangkat lunak adalah suatu proses pengorganisasian kumpulan metode dan konvensi notasi yang telah didefinisikan untuk mengembangkan perangkat lunak.

Pada penelitian ini, penulis menggunakan metode waterfall dalam pengembangan perangkat lunak. Berikut adalah tahapan – tahapan dalam metode waterfall :



Gambar 3.2 Metode Waterfall

Metode Waterfall memiliki tahapan-tahapan sebagai berikut :

1. Analisa Kebutuhan

Kondisi, kriteria, syarat atau kemampuan yang harus dimiliki oleh perangkat lunak untuk memenuhi apa yang disyaratkan atau diinginkan pengguna.

1. Desain Sistem

Tahapan perancangan sistem mengalokasikan kebutuhan-kebutuhan sistem baik perangkat keras maupun perangkat lunak dengan membentuk arsitektur sistem secara keseluruhan. Perancangan perangkat lunak melibatkan identifikasi dan penggambaran abstraksi sistem dasar perangkat lunak dan hubungannya.

1. Implementasi

Tahap ini merupakan tahap setelah dilakukannya perancangan sistem. Perancangan yang telah dibuat akan diterjemahkan menjadi kode pemrograman dengan menggunakan bahasa php berbasis website.

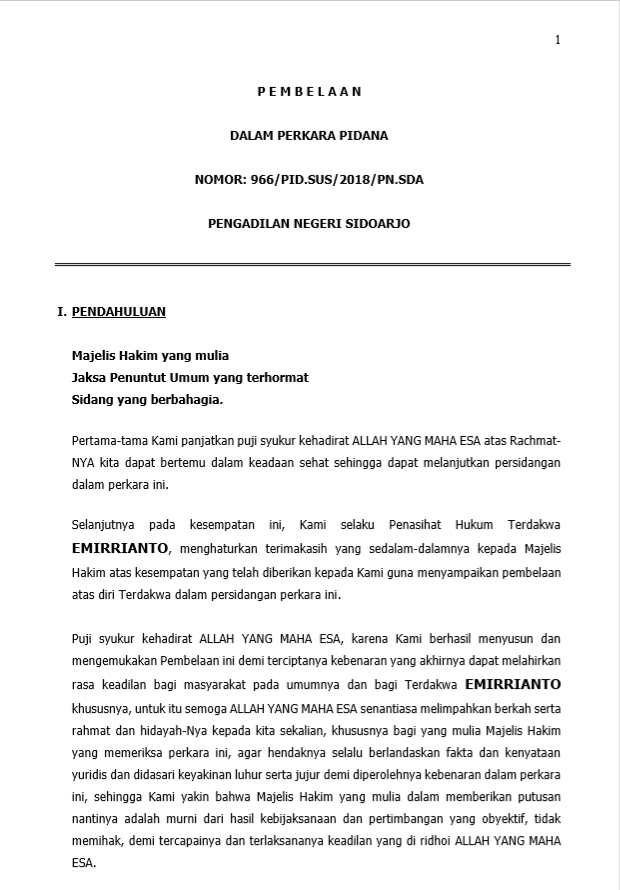
1. Pengujian Unit

Untuk pengujian unit digunakan *black box testing. Black Box Testing* berfokus pada spesifikasi fungsional dari perangkat lunak. Pengujian yang dilakukan masing-masing unit. Setelah integrasi seluruh sistem diuji untuk mengecek  setiap kegagalan maupun kesalahan.

1. Pemeliharaan

Pada tahap pemeliharaan sistem, aplikasi yang telah dijalankan akan di lakukan perbaikan jika terdapat kesalahan yang tidak ditemukan pada langkah sebelumnya. Perbaikan implementasi unit sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

## Metode Pengolahan Data

Data yang telah diperoleh perlu dilakukan pengolahan agar dapat dilakukan proses peringkasan teks otomatis. Peringkasan otomatis dilakukan dengan menerapkan metode LSA yang memiliki tahapan yaitu untuk menghilangkan *noise* dilakukan perhitungan menggunakan metode SVD. Dari hasil tersebut, akan dilakukan penyeleksian kalimat yang memiliki skor tertinggi. Di bawah ini merupakan contoh data yang akan digunakan dalam peringkasan dokumen :

Gambar 3.3 Contoh Data : Nota Pembelaan

Data atau kalimat yang akan digunakan untuk implementasi manual metode preprocessing dan perhitungan metode pengambilan ringkasan ditunjukan pada paragraf dibawah ini.

“Emmirianto memosting status tersebut hanya dalam waktu kurang dari 1 hari dan belum berdampak bagi masyarakat. Lalu ia melakukan perbuatan tersebut atas dasar keinginan sendiri dan hal tersebut merupakan bentuk emosi dari terdakwa yang akhirnya mengalahkan akal sehatnya. Sehingga terdakwa tidak kontrol dan merasa jengkel.”

1. Pre-Processing

Pada tahapan ini, data tekstual akan diubah menjadi teks agar dapat diolah oleh sistem. Penelitian ini menggunakan dokumen sebagai inputan awal. Proses yang digunakan antara lain :

1. Pembentukan Kalimat

Pembentukan kalimat yaitu pemecahan teks dokumen menjadi kumpulan kalimat berdasarkan delimiter.

Tabel 3.1 Pembentukan Kalimat

|  |  |
| --- | --- |
| No | Pembentukan Kalimat |
| 1 | Emmirianto memosting status tersebut hanya dalam waktu kurang dari 1 hari dan belum berdampak bagi masyarakat. |
| 2 | Lalu ia melakukan perbuatan tersebut atas dasar keinginan sendiri dan hal tersebut merupakan bentuk emosi dari terdakwa yang akhirnya mengalahkan akal sehatnya. |
| 3 | Sehingga Terdakwa tidak kontrol dan merasa jengkel. |

1. *Case Folding*

*Case folding* merupakan pengubahan huruf pada kalimat menjadi huruf kecil (*lowercase*) dan penghilangan karakter yang tidak valid seperti tanda baca.

Tabel 3.2 Case Folding

|  |  |
| --- | --- |
| No | *Case Folding* |
| 1 | emmirianto memosting status tersebut hanya dalam waktu kurang dari hari dan belum berdampak bagi masyarakat |
| 2 | lalu ia melakukan perbuatan tersebut atas dasar keinginan sendiri dan hal tersebut merupakan bentuk emosi dari terdakwa yang akhirnya mengalahkan akal sehatnya |
| 3 | sehingga terdakwa tidak kontrol dan merasa jengkel |

1. *Tokenizing*

Pada proses ini, kalimat tersebut dipecah kembali menjadi beberapa kata tunggal penyusunnya.

Tabel 3.3 Tokenizing

|  |  |  |  |
| --- | --- | --- | --- |
| *Tokenizing* | | | |
| emmirianto | belum | keinginan | mengalahkan |
| memosting | berdampak | sendiri | akal |
| status | bagi | dan | sehatnya |
| tersebut | masyarakat | hal | sehingga |
| hanya | lalu | tersebut | terdakwa |
| dalam | Ia | bentuk | tidak |
| waktu | melakukan | emosi | kontrol |
| kurang | perbuatan | dari | dan |
| dari | tersebut | terdakwa | merasa |
| hari | atas | yang | jengkel |
| dan | dasar | akhirnya | Dll. |

1. *Stopword Removal*

*Stopword removal* adalah proses penghilangan kata-kata yang tidak merepresentasikan isi dokumen.

Tabel 3.4 Stopword Removal

|  |  |  |  |
| --- | --- | --- | --- |
| *Stopword Removal* | | | |
| emmirianto | berdampak | dasar | emosi |
| memosting | masyarakat | bentuk | terdakwa |
| status | perbuatan | mengalahkan | Dll. |

1. *Stemming*

*Stemming* adalah proses pengembalian kata tunggal yang memiliki imbuhan menjadi kata dasar.

Tabel 3.5 Stemming

|  |  |  |  |
| --- | --- | --- | --- |
| *Stemming* | | | |
| emmirianto | dampak | dasar | emosi |
| posting | masyarakat | bentuk | dakwa |
| status | buat | kalah | Dll. |

1. Processing
2. TF-IDF

Setelah dokumen diproses dengan cara *preprocessing, tokenizing, filtering* dan *stemming*, selanjutnya dilakukan proses pembobotan kata. Pada Metode ini pembobotan kata dalam sebuah dokumen dilakukan dengan mengalikan nilai TF dan IDF.

Tabel 3.6 Contoh Perhitungan DF

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Kata penting | Frekuensi Kata pada Kalimat | | | | df |
| **Kalimat 1** | **Kalimat 2** | **Kalimat 3** |  | |
| emmirianto | 1 | 1 | 0 | 2 | |
| posting | 1 | 0 | 0 | 1 | |
| status | 1 | 0 | 0 | 1 | |
| 1 | 1 | 0 | 0 | 1 | |
| dampak | 1 | 0 | 0 | 1 | |
| masyarakat | 1 | 0 | 0 | 1 | |
| buat | 0 | 1 | 0 | 1 | |
| dasar | 0 | 1 | 0 | 1 | |
| bentuk | 0 | 1 | 0 | 1 | |
| emosi | 0 | 1 | 1 | 2 | |
| dakwa | 0 | 1 | 0 | 1 | |
| kalah | 0 | 1 | 0 | 1 | |
| sehat | 0 | 1 | 0 | 1 | |
| kontrol | 0 | 0 | 1 | 1 | |
| jengkel | 0 | 0 | 1 | 1 | |

Tabel 3.7 Contoh Perhitungan TF-IDF

|  |  |  |
| --- | --- | --- |
| W | | |
| tf\*idf | | |
| 1.176091 | 1.176091 | 0 |
| 1.477121 | 0 | 0 |
| 1.477121 | 0 | 0 |
| 1.477121 | 0 | 0 |
| 1.477121 | 0 | 0 |
| 1.477121 | 0 | 0 |
| 0 | 1.477121 | 0 |
| 0 | 1.477121 | 0 |
| 0 | 1.477121 | 0 |
| 0 | 1.477121 | 0 |
| 0 | 1.176091 | 1.176091 |
| 0 | 1.477121 | 0 |
| 0 | 1.477121 | 0 |
| 0 | 0 | 1.477121 |
| 0 | 0 | 1.477121 |

1. Latent Semantic Analysis

Adapun langkah-langkah LSA sebagai berikut (Mandar & Gunawan, 2017):

1. Membentuk matriks Amn.

𝐴 = 𝑈𝑆𝑉𝑇 (3.1)

A adalah matriks dokumen yang mewakili kalimat atau kata yang dikenal dengan matriks Amn. Dari perhitungan TF-IDF, dilakukan pembentukan matriks A dengan cara memasukkan nilai TF-IDF. Selanjutnya dilakukan perhitungan menggunakan metode *Singular Value Decomposition* (SVD). Seperti pada rumus 7.2. Akan didapat maktriks U mendiskripsikan matriks orthogonal 𝑚 × 𝑚 yang dikenal dengan istilah *Left Singular Vector* *. Right Singular Vektor* (V) merupakan matriks orthogonal 𝑛 × 𝑛 yang diperoleh dari *eigen vector* matriks ATA, sedangkan matriks diagonal S dihasilkan dari *eigen value* matriks ATA yang diakarkan.

Tabel 3.8 Hasil Perhitungan Matriks ATA (X)

|  |  |  |
| --- | --- | --- |
| Kalimat 1 | Kalimat 2 | Kalimat 3 |
| 12,29262666 | 1,38319065 | 0 |
| 1,38319065 | 15,85770451 | 1,38319065 |
| 0 | 1,38319065 | 5,746965052 |

1. Perhitungan *Eigen Value*

Setelah dilakukan perhitungan ATA, dilanjutkan dengan melakukan perhitungan *eigen value*. Dengan Rumus :

det |X-λI| (3.2)

Tabel 3.9 Hasil Perhitungan X-λI

|  |  |  |
| --- | --- | --- |
| Kalimat 1 | Kalimat 2 | Kalimat 3 |

|  |  |  |
| --- | --- | --- |
| 12,2926266553501 - λ | 1,38319065 | 0 |
| 1,38319065 | 15,8577045061218 - λ | 1,38319065 |
| 0 | 1,38319065 | 5,74696505191634 - λ |

Polinomial yang didapat dari det |X-λI| yaitu -λ3 + 33,8973 λ2 – 352,885 λ + 1085,76. Dengan akar akar persamaan sebagai berikut :

Tabel 3.10 Hasil Perhitungan det |X-λI| (Eigen Value)

|  |  |
| --- | --- |
| λ | Hasil |
| λ1 | 16,491 |
| λ2 | 11,85 |
| λ3 | 5,556 |

Dari masing-masing *eigen value* dilakukan perhitungan untuk mencari *eigen vector*-nya dan dilanjutkan dengan normalisasi. Rumus normalisasi *eigen vector* :

===  (3.3)

Dari hasil normalisasi tersebut disatukan membentuk matriks V dan di *transpose*-kan. Seperti contoh dibawah ini.

Tabel 3.11 Matriks VT

|  |  |  |
| --- | --- | --- |
| Kalimat 1 | Kalimat 2 | Kalimat 3 |
| 0,310572622 | 0,942769607 | 0,121367684 |
| -0,950134866 | 0,304126009 | 0,068928272 |
| 0,028072398 | -0,136722469 | 0,990211547 |

1. Membentuk matriks S

Pembentukan mastriks S yaitu dengan cara mengurutkan nilai tertinggi *eigen value* kemudian diakarkan. Hasil matriks dapat dilihat pada tabel dibawah ini.

Tabel 3.12 Matriks S

|  |  |  |
| --- | --- | --- |
| Matriks S | | |
| 4,060911228 | 0 | 0 |
| 0 | 3,442382896 | 0 |
| 0 | 0 | 2,357116883 |

1. Hasil ringkasan berdasarkan perhitungan *length*

Hasil ringkasan ditentukan berdasarkan skor tertinggi dari perhitungan *length* pada setiap nilai matriks VT dengan menggunakan rumus :

Sk = (3.4)

Di mana 𝑆𝑘 adalah panjang vektor k pada kalimat yang dimodifikasi oleh laten vektor. n adalah jumlah ruang demensi baru. Hasil dari length terbesar pada setiap dokumen kalimat akan dijadikan ringkasan. Hasil dari perhitungan dapat dilihat pada tabel dibawah ini.

Tabel 3.13 Hasil skor perhitungan Sk

|  |  |
| --- | --- |
|  | Hasil Skor |
| Kalimat 1 | 1,575819146 |
| Kalimat 2 | **1,954901801** |
| Kalimat 3 | 1,668110957 |

Didapatkan hasil bahwa kalimat 2 akan dijadikan sebagai ringkasan dikarenakan memiliki hasil skor tertinggi.

## Metode Pengujian

Untuk pengujian pada ringkasan dapat dilakukan dengan mencari nilai *recall*, *precision* dan *f-measure*. *Recall* ialah kemampuan untuk mengambil peringkat teratas yang sebagian besar relevan (benar). *Precision* adalah berapa banyak dokumen yang berhasil diambil oleh sistem, sedangkan untuk mengukur kualitas *recall* dan *precision* menggunakan *f-measure*.

(3.5)

(3.6)

(3.7)

## Analisa Hasil dan Kesimpulan

Proses pengambilan kesimpulan yaitu dengan cara melihat akurasi dari hasil peringkasan teks otomatis pada dokumen hukum.

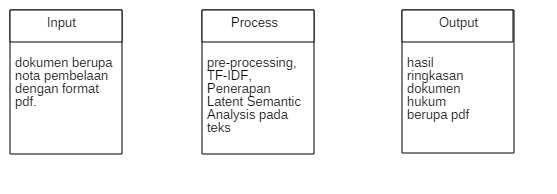
# BAB IV ANALISA DAN PERANCANGAN

Bab analisis dan perancangan menjelaskan tentang analisis dan perancangan yang digunakan dalam mengimplementasikan metode di sistem peringkasan teks pada dokumen hukum secara otomatis.

## Analisis Sistem

Sistem yang dirancang dan dibangun dalam penelitian ini adalah sistem peringkasan dokumen hukum otomatis menggunakan metode LSA yang digunakan untuk memudahkan pengguna dalam meringkas dokumen dalam jumlah besar. Pengguna dari sistem ini adalah pihak aparat hukum seperti pengacara pada Kantor Advokat Ahmad Riyadh Umar Balhmar, Surabaya, Jawa Timur, Indonesia.

. Diagram blok perancangan terdapat input, proses, dan output dimana input dari sistem ini adalah nota pembelaan yang memiliki format PDF dan akan diubah menjadi format yang dapat terbaca oleh sistem. Kemudian diproses, dari data yang telah diolah tersebut akan dilakukan perhitungan TFIDF untuk kesamaan kata dan dilakukan perhitungan LSA untuk mendapatkan output berupa hasil peringkasan dokumen. Berikut adalah diagram blok terdapat pada Gambar 4.1 :



Gambar 4.1 Diagram Blok Perancangan

### Analisis Kebutuhan Fungsional

Kebutuhan fungsional adalah kebutuhan yang berisi proses-proses apa saja/ layanan apa saja yang nantinya dilakukan oleh sistem. Berikut analisis kebutuhan fungsional pada sistem yang akan dibangun:

1. Admin dapat mengelola data pengguna lain.
2. Admin dan pengguna lain dapat melakukan login.
3. Admin dan pengguna lain dapat mengelola dokumen.
4. Admin dan pengguna lain dapat melakukan peringkasan.
5. Admin dan pengguna lain dapat melihat hasil peringkasan.
6. Admin dan pengguna lain dapat mengunduh hasil peringkasan.
7. Admin dapat melakukan pengujian.
8. Admin dapat melihat grafik hasil pengujian.
9. Admin dan pengguna lain dapat melakukan logout.

### Kebutuhan *Non*-fungsional

Kebutuhan *Non*-fungsional adalah kebutuhan yang tidak secara langsung terkait dengan fitur tertentu di dalam sistem. Berikut analisis kebutuhan *non*-fungsional pada sistem yang akan dibangun:

Kebutuhan Perangkat Keras

Kebutuhan perangkat keras yang dibutuhkan untuk penelitian yaitu pada Tabel 4.1 berikut :

Tabel 4.1Kebutuhan Perangkat Keras

|  |  |
| --- | --- |
| Nama Komponen | Spesifikasi |
| Processor | Intel Inside Core i3 [@2.00GHz](mailto:@2.00GHz) |
| RAM (Memory) | 8GB |
| Hardisk | 500GB |

Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang dibutuhkan untuk penelitian yaitu pada Tabel 4.2 sebagai berikut:

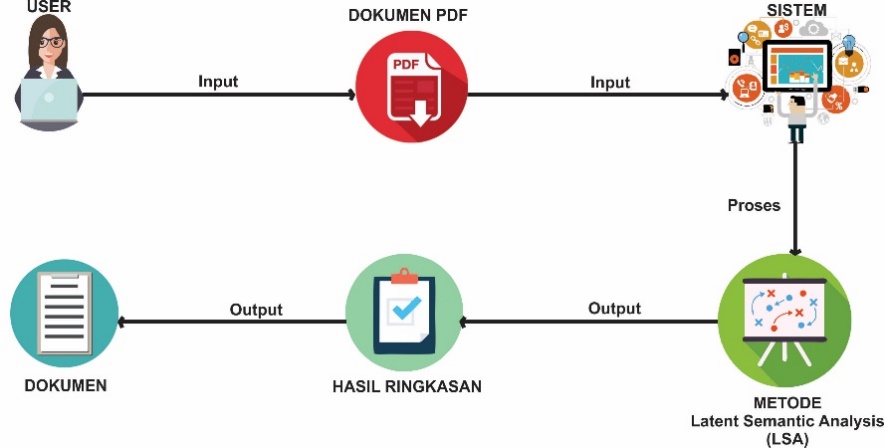
Tabel 4.2 Kebutuhan Perangkat Lunak

|  |  |
| --- | --- |
| Nama Komponen | Spesifikasi |
| Sistem Operasi | Microsoft Windows 8.1 |
| Editor Pemrograman | Sublime Text |
| Web Server | Xampp 3.2 |
| Web Browser | Google Crome |
| Codeigniter | Codeigniter 3.1.10 |
| Database | MySQL |
| PHP | PHP versi 7.2 |

## Perancangan Sistem

1. Desain Sistem

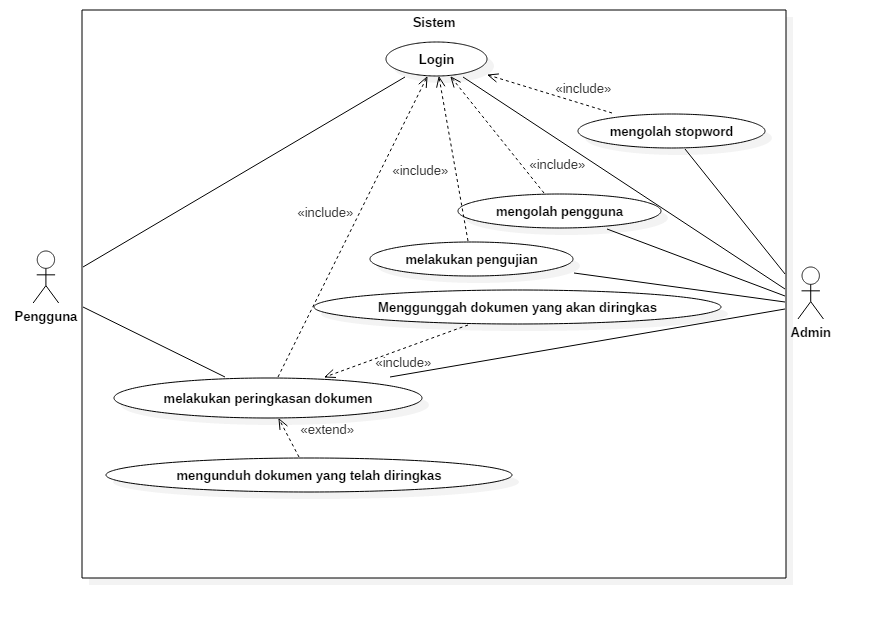
Tujuan dari sistem ini untuk melakukan peringkasan dokumen hukum secara otomatis agar memudahkan pengguna dalam meringkas dokumen dalam jumlah besar . Pada Gambar 4.1 sebagai berikut :



Gambar 4.2 Desain Sistem

### *Use Case* Diagram

*Use Case* adalah gambaran fungsionalitas dari suatu sistem, sehingga pengguna sistem dapat memahami dan mengerti mengenai kegunaan sistem. Di bawah ini merupakan *use case* diagram dari sistem yang akan dibangun :



Gambar 4.3 Use Case Diagram

Dapat dilihat pada Gambar 4.3, pengguna dan admin dapat melakukan peringkasan dokumen dimana harus meginputkan dokumen yang akan diringkas dan melakukan *pre-processing*. Sedangkan untuk mengunduh dokumen yang telah diringkas tidak diharuskan, pengguna dan admin dapat melihat langsung pada aplikasi hasil ringkasan tersebut. Admin dapat melakukan pengujian ringkasan yang telah dilakukan oleh sistem. Admin juga dapat mengolah pengguna seperti menambahkan dan mengubah data pengguna.

### *Use Case* Spesifikasi

Deskripsi dari *use case* diagram yang menjelaskan bagaimana aktor dan *use* *case* terlibat dalam sistem. Berikut ini merupakan spesifikasi *use case* dari sistem yang akan dibangun :

Tabel 4.3 Spesifikasi Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| No | Aktor | *Use Case* | Keterangan |
| 1. | Admin, Pengguna lain | Mengolah dokumen | Admin dan pengguna lain dapat melihat dan menghapus dokumen. |
| 2. | Admin, Pengguna lain | Melakukan peringkasan | Admin dan pengguna lain dapat melakukan peringkasan dengan cara menghitung nilai semanti*k* kalimat menggunakan metode LSA*.* |
| 3. | Admin | Melakukan pengujian | Admin dapat melakukan pengujian dengan melakukan export sample, lalu melakukan peringkasan manual melalui file tersebut. Setelah itu melakukan import pengujian. Hasil perhitungan akurasi akan tampil setelah melakukan import. |
| 4. | Admin | Mengolah pengguna | Menampilkan, menambahkan dan mematikan akun pengguna serta mengubah data pengguna. |
| 5. | Admin | Mengolah Stopword | Menampilkan, menambahkan dan menghapus *stopword* |

### *Use Case* Skenario

Untuk lebih memperjelas use case pada Table 4.3, Berikut ini skenario (*flow of event*) untuk beberapa *use case* utama, yang menggambarkan interaksi aktor dengan use case tersebut.

Tabel 4.4 Skenario - mengolah dokumen

|  |  |
| --- | --- |
| Mengolah dokumen | |
| Tujuan | Menampilkan dan menghapus dokumen |
| Aktor | Admin dan Pengguna lain |
| Kondisi awal | 1. Login |
| Skenario utama | 1. Aktor memilih menu *document*. 2. Aktor memilih sub-menu *view all*. 3. Sistem menampilkan semua dokumen yang tersimpan. |
| Skenario alternatif | 1. Jika aktor ingin menghapus dokumen, aktor diharuskan untuk memilih data yang akan dihapus terlebih dahulu dan dilanjutkan dengan menekan tombol *delete*. 2. Jika aktor ingin melihat isi dokumen, aktor diharuskan untuk memilih dokumen yang akan dlihat terlebih dahulu dan dilanjutkan dengan menekan tombolyang ada pada kolom “nama dokumen”. |
| Kondisi Akhir | Sistem menampilkan dokumen yang tersimpan dalam *database*. |

Tabel 4.5 Skenario - mengolah stopword

|  |  |
| --- | --- |
| Mengolah *stopword* | |
| Tujuan | Menampilkan, menambahkan dan menghapus *stopword* |
| Aktor | Admin |
| Kondisi awal | 1. Login sebagai admin. |
| Skenario utama | 1. Admin memilih menu *stopword*. 2. Sistem menampilkan data *stopword*. |
| Skenario alternatif | 1. Jika Admin ingin menambahkan data *stopword*, admin dapat menekan tombol tambah. 2. Jika Admin ingin menghapus data *stopword*, admin diharuskan untuk memilih data *stopword* yang akan dihapus terlebih dahulu dan dilanjutkan dengan menekan tombol *delete*. |
| Kondisi Akhir | Sistem menampilkan data *stopword* sesuai yang tersimpan dalam *database*. |

Tabel 4.6 Skenario – melakukan peringkasan

|  |  |
| --- | --- |
| Melakukan peringkasan | |
| Tujuan | Mendapatkan dokumen yang telah diringkas |
| Aktor | Admin, Pengguna Lain |
| Kondisi awal | 1. Login 2. Memiliki dokumen yang akan diringkas dengan format pdf |
| Skenario utama | 1. Aktor memilih menu *document*. 2. Aktor memilih sub-menu *upload and summarize.* 3. Aktor mengunggah dokumen yang akan diringkas. 4. Aktor memilih tingkat kompresi panjangnya suatu ringkasan*.* 5. Sistem melakukan perhitungan metode LSA. 6. Sistem menampilkan proses perhitungan. 7. Sistem menampilkan dokumen yang sudah diringkas. 8. Aktor mengunduh hasil peringkasan dokumen tersebut. |
| Skenario alternatif | 1. Aktor memilih menu sub-menu upload and summarize*.* 2. Aktor mengunduh hasil peringkasan dokumen. |
| Kondisi Akhir | Sistem mendapatkan hasil perhitungan dari penerapan metode LSA dan menampilkan dokumen yang telah diringkas. |

Tabel 4.7 Skenario – melakukan pengujian

|  |  |
| --- | --- |
| Melakukan pengujian | |
| Tujuan | Mendapat tingkat akurasi dari metode Latent Semantic Analysis |
| Aktor | Admin |
| Kondisi awal | 1. Login sebagai admin 2. Terdapat dokumen yang sudah diringkas secara manual berupa pdf. |
| Skenario utama | 1. Admin memilih menu pengujian. 2. Admin menekan tombol tambah pengujian. 3. Admin memilih dokumen untuk dijadikan uji coba. 4. Admin mengunggah dokumen (pdf) yang telah diringkas secara manual sesuai dengan dokumen uji coba yang telah dipilih. 5. Sistem menampilkan hasil perhitungan *precision*, *recall*, *f*-*measure*. |
| Skenario alternatif | 1. Admin dapat memilih dokumen secara acak dengan menginputkan dokumen yang diinginkan. |
| Kondisi Akhir | Sistem menampilkan data penggunasesuai yang tersimpan dalam *database*. |

Tabel 4.8 Skenario – mengolah pengguna

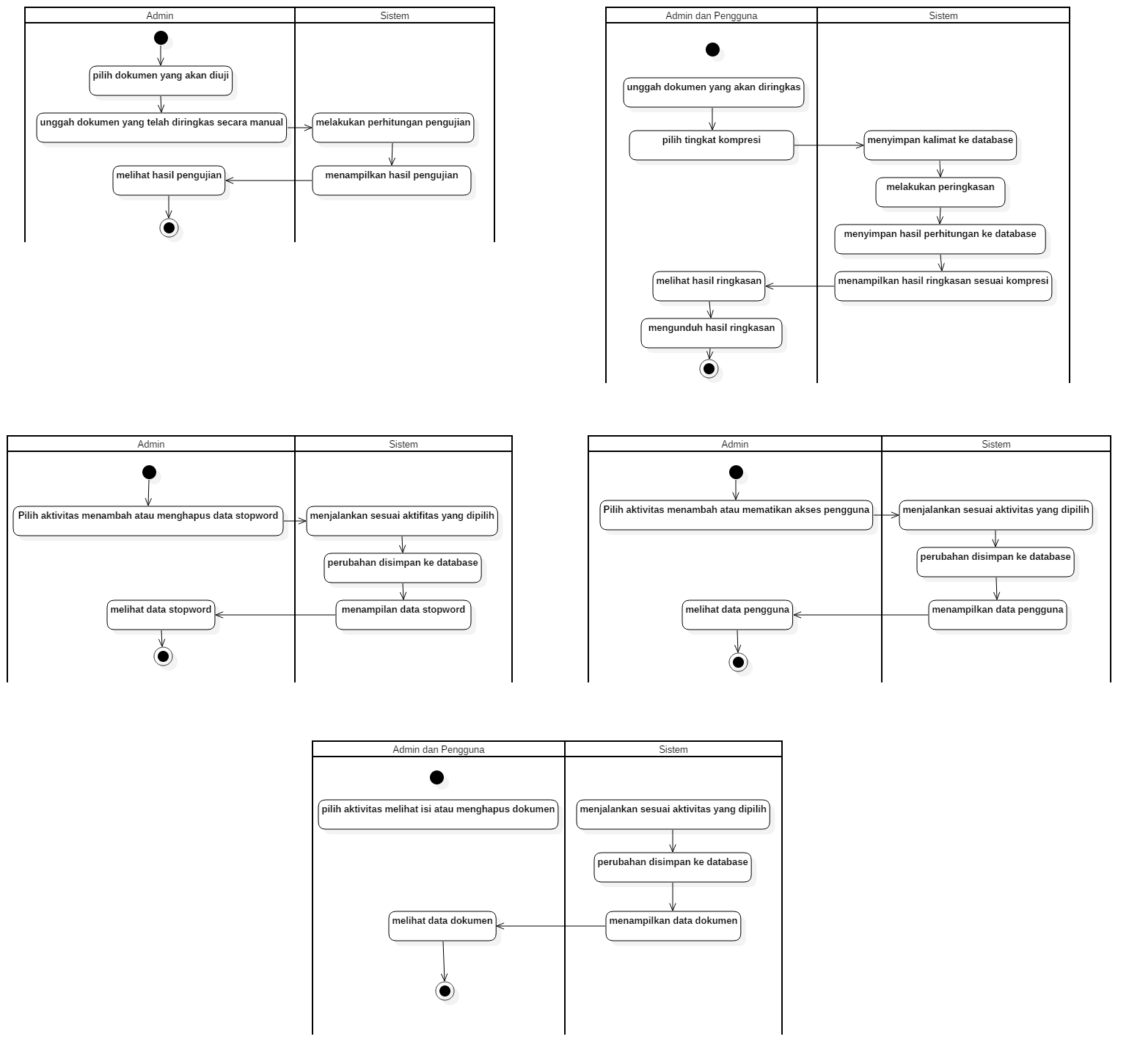
|  |  |
| --- | --- |
| Mengolah pengguna | |
| Tujuan | Menampilkan, menambahkan dan mematikan akun pengguna serta mengubah data pengguna. |
| Aktor | Admin |
| Kondisi awal | 1. Login sebagai admin. 2. Sistem menampilkan data pengguna. |
| Skenario utama | 1. Admin memilih menu pengguna. 2. Sistem menampilkan datapengguna. |
| Skenario alternatif | 1. Jika admin ingin mematikan akun penggua, maka admin diharuskan memilih pengguna yang tersebut dan dilanjutkan dengan menekan tombol matikan akun. 2. Jika admin ingin menambahkan pengguna agar dapat login ke sistem, maka admin dapat menambahkan pengguna tersebut dengan menekan tambah pengguna. 3. Jika pengguna lain lupa *password* atau *username*, maka admin dapat mengubah data pengguna agar dapat *login* kembali. |
| Kondisi Akhir | Admin dapat melihat data pengguna yang tersimpan di *database* |

### *Activity* Diagram

Berikut adalah *activity* diagram yang dirancang untuk untuk menjelaskan alur proses antara aktor dan sistem :

1. Alur Pengolahan Dokumen

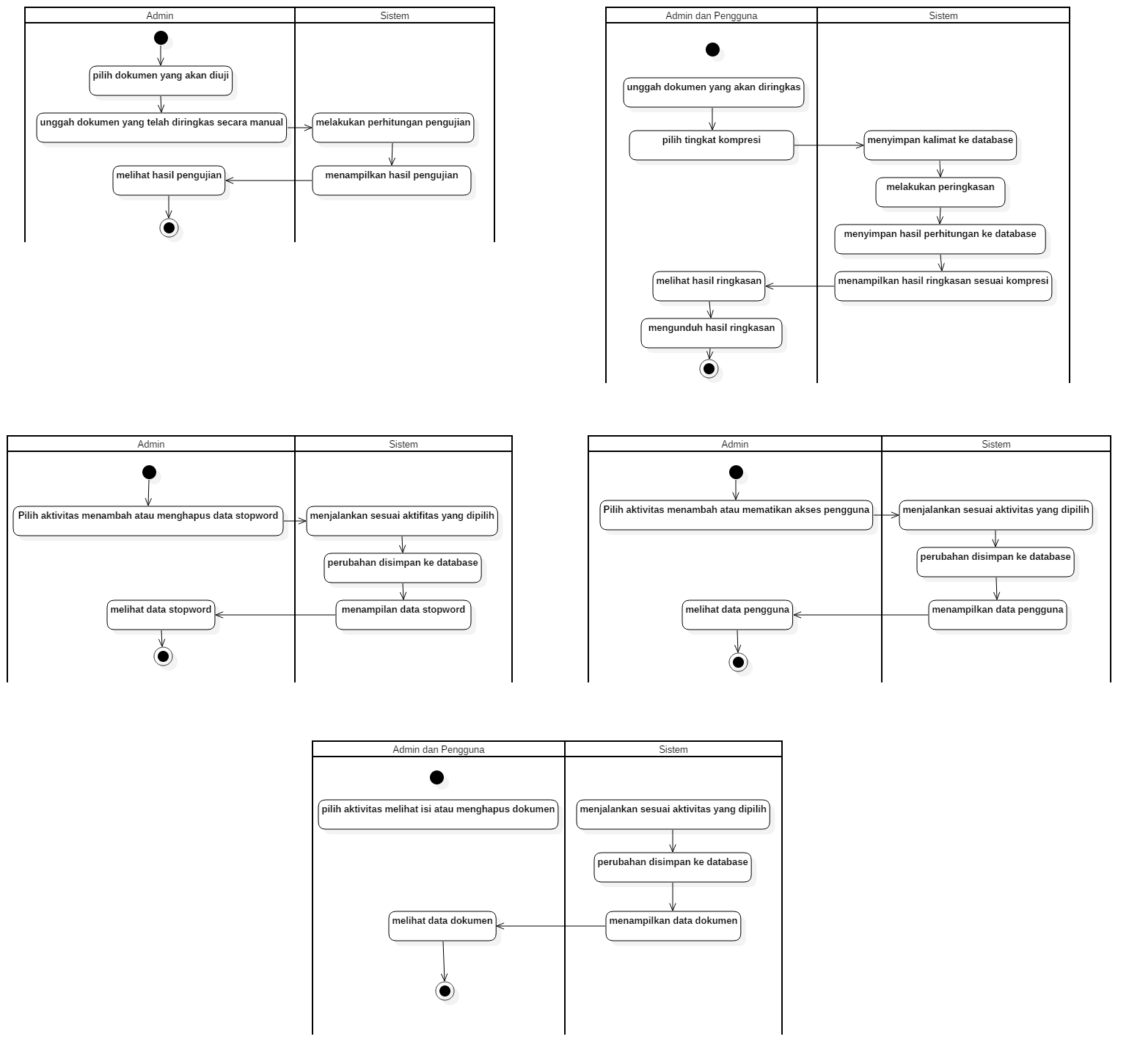
Pada proses pengolahan dokumen, admin dan pengguna dapat memilih untuk melihat isi atau menghapus dokumen. Setelah itu sistem akan menjalankan sesuai dengan aktivitas yang dipilih. Jika terdapat perubahan, maka *database* akan menyimpan sesuai dengan perubahan yang terjadi. *Activity* diagram pengolahan dokumen ditunjukan pada Gambar 4.4.



Gambar 4.4 Activity Diagram Pengolahan Dokumen

1. Alur Pengolahan *Stopword*

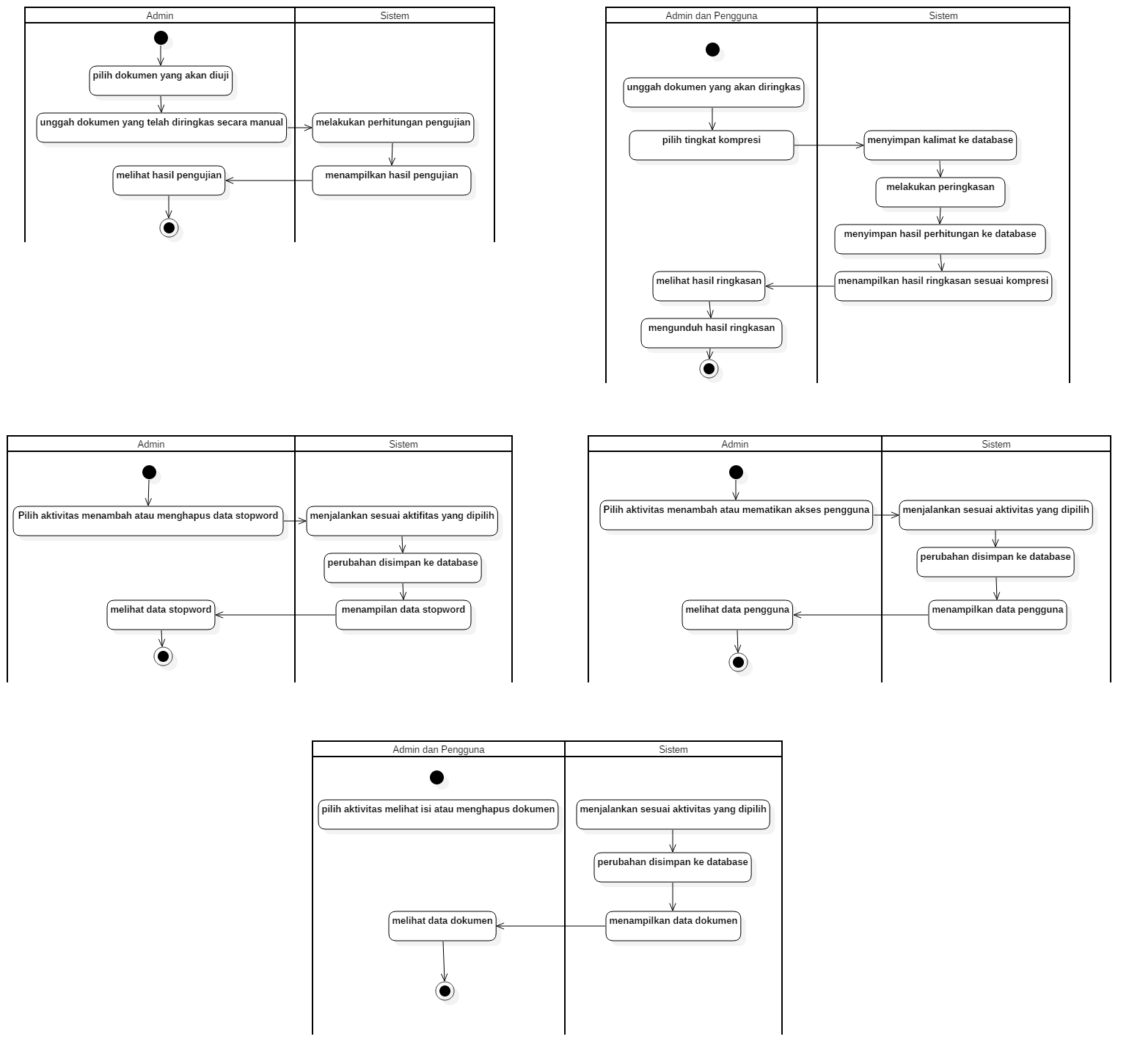
Pada proses pengolahan *stopword*, admin dapat memilih untuk menambah atau menghapus data *stopword*. *Stopword* adalah kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna. Setelah memilih aktivitas yang akan dilakukan, sistem akan berjalan sesuai yang dipilih. Jika terdapat perubahan, maka *database* akan menyimpan perubahan yang terjadi. *Activity* diagram pengolahan *stopword* ditunjukan pada Gambar 4.5.



Gambar 4.5 Activity Diagram Pengolahan Data Stopword

1. Alur Pengolahan Pengguna

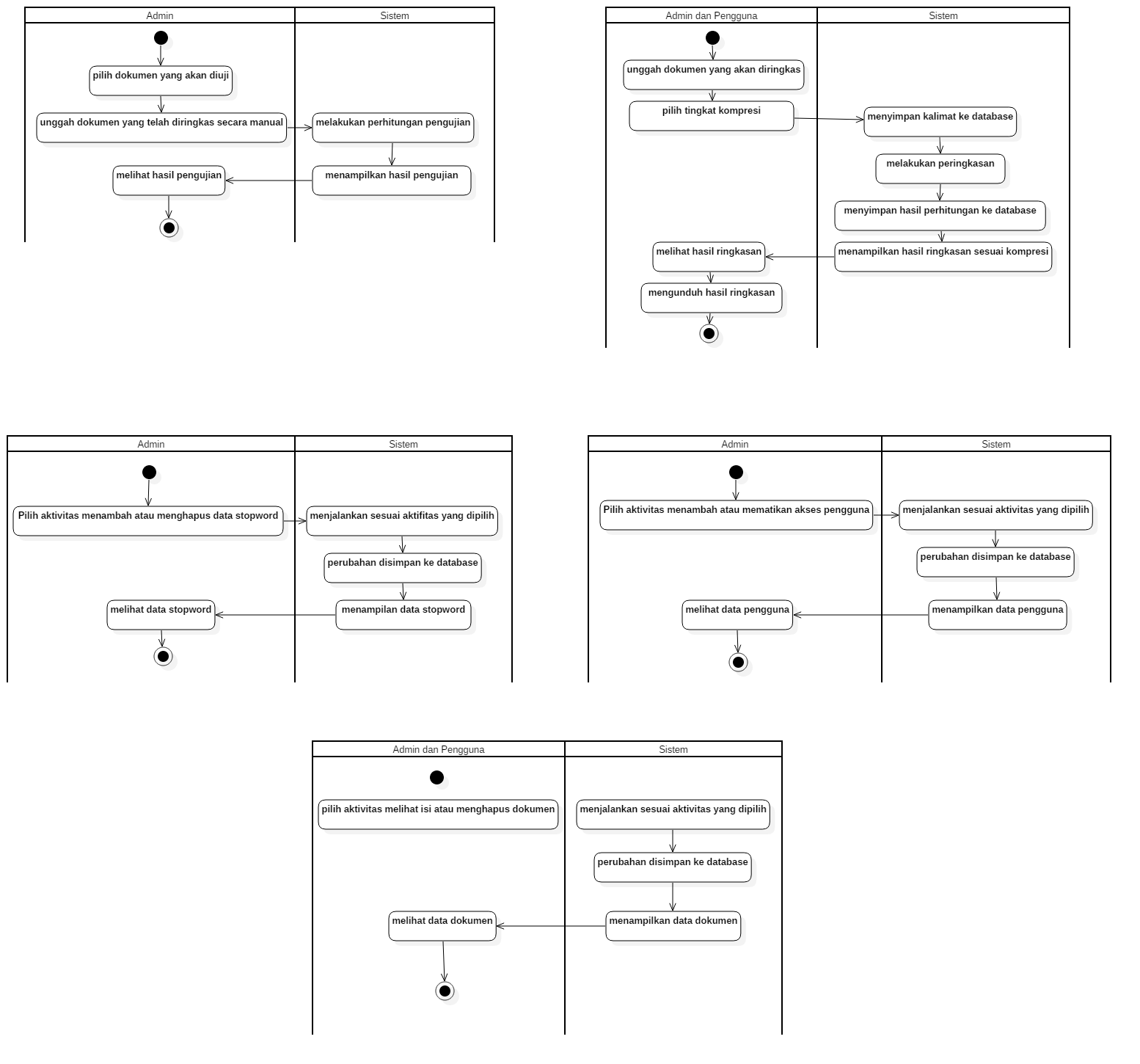
Pada proses pengolahan data pengguna, admin dapat memilih untuk menambah atau mematikan akses pengguna. Setelah memilih aktivitas yang akan dilakukan, sistem akan berjalan sesuai yang dipilih. Jika terdapat perubahan, maka *database* akan menyimpan perubahan yang terjadi. *Activity* diagram pengolahan data pengguna ditunjukan pada Gambar 4.6.



Gambar 4.6 Activity Diagram Pengolahan Data Pengguna

1. Alur Peringkasan Dokumen

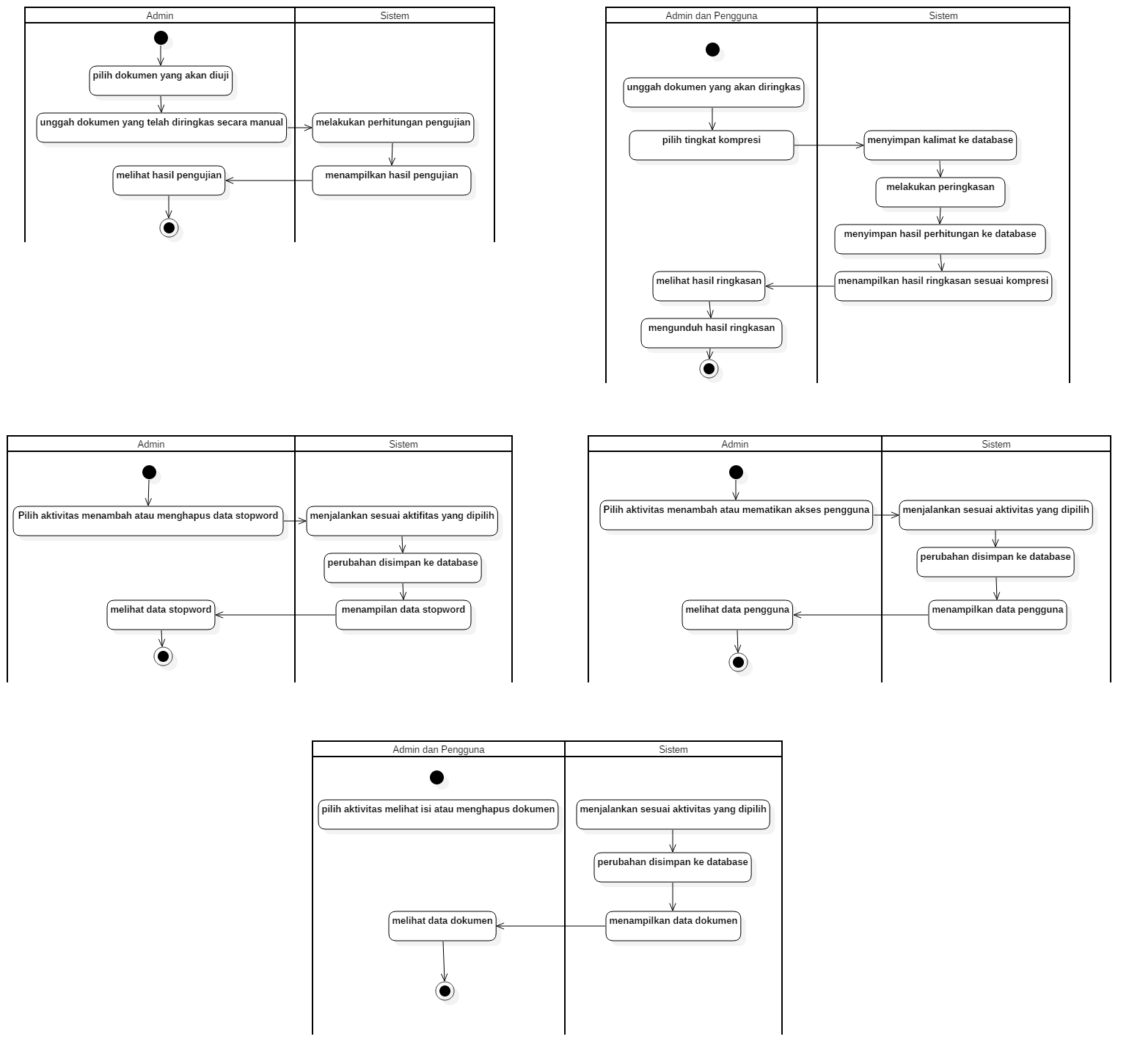
Pada proses melakukan peringkasan dimulai dengan admin dan pengguna mengunggah dokumen yang akan diringkas. Sebelum dilakukan peringkasan, admin dan pengguna memilih tingkat kompresi panjangnya hasil ringkasan. Selanjutnya, sistem melakukan perhitungan metode LSA dan hasil dari perhitungan tersebut akan disimpan ke dalam database. Hasil peringkasan diurutkan berdasarkan hasil perhitungan tertinggi, Sedangkan panjang peringkasan sesuai dengan kompresi yang telah dipilih. *Activity* diagram melakukan peringkasan ditunjukan pada Gambar 4.7



Gambar 4.7 Activity Diagram Peringkasan Dokumen

1. Alur Pengujian Ringkasan

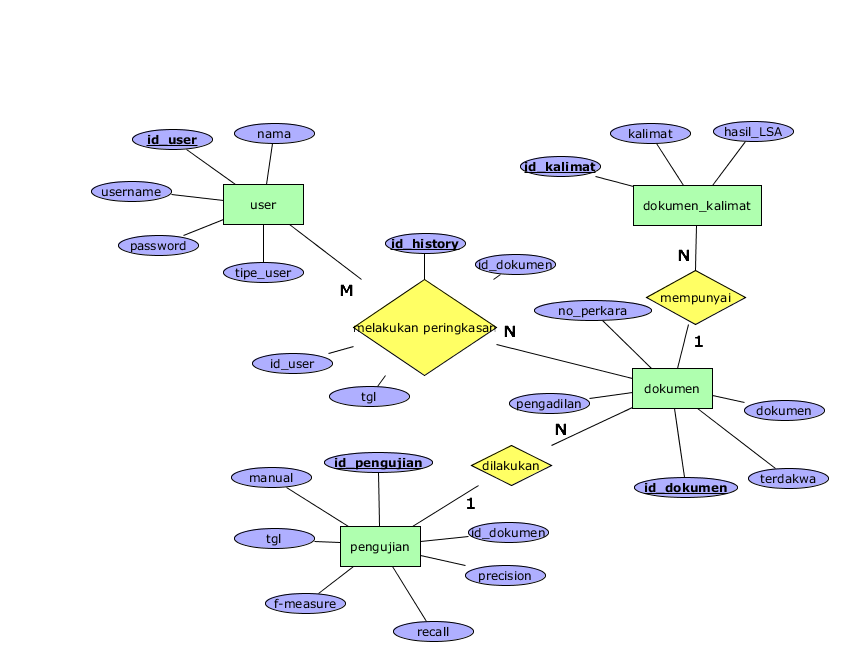
Pengujian dilakukan diawali dengan admin memilih dokumen yang ingin diuji dan melakukan pengunggahan hasil ringkasan dokumen secara manual. Kemudian sistem melakukan perhitungan pengujian. Setelah itu hasil dari perhitungan tersebut ditampilkan pada halaman web. Diagram aktifitas melakukan pengujian ditunjukan pada Gambar 4.8



Gambar 4.8 Activity Diagram Pengujian Peringkasan

### ERD

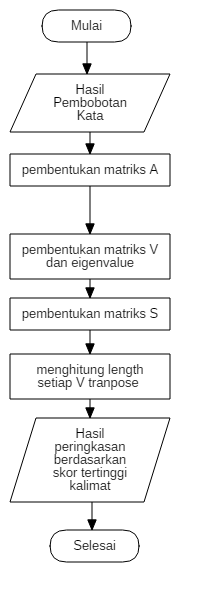
*Entity Relationship Diagram* (ERD) adalah suatu diagram yang digunakan untuk merancang suatu basis data. ERD dapat dipergunakan untuk memperlihatkan hubungan atau relasi antar entitas atau objek yang terlihat beserta atributnya. ERD bertujuan untuk membantu dalam merancang relasi antar tabel dalam membuat database. Pada sistem ini, ERD ditunjukan pada Gambar 4.9



Gambar 4.9 Entity Relationship Diagram

Pada gambar diatas dijelaskan bahwa banyak user dapat melakukan peringkasan banyak dokumen. Dikarenakan *many* *to* *many* maka menghasilkan tabel baru yang akan diberi nama tabel *history*. Selanjutnya satu dokumen bisa memiliki beberapa kalimat. Kemudian satu pengujian dapat dilakukan ke beberapa dokumen.

### Alur Metode Latent Semantic Analysis



Gambar 4.10 Alur LSA

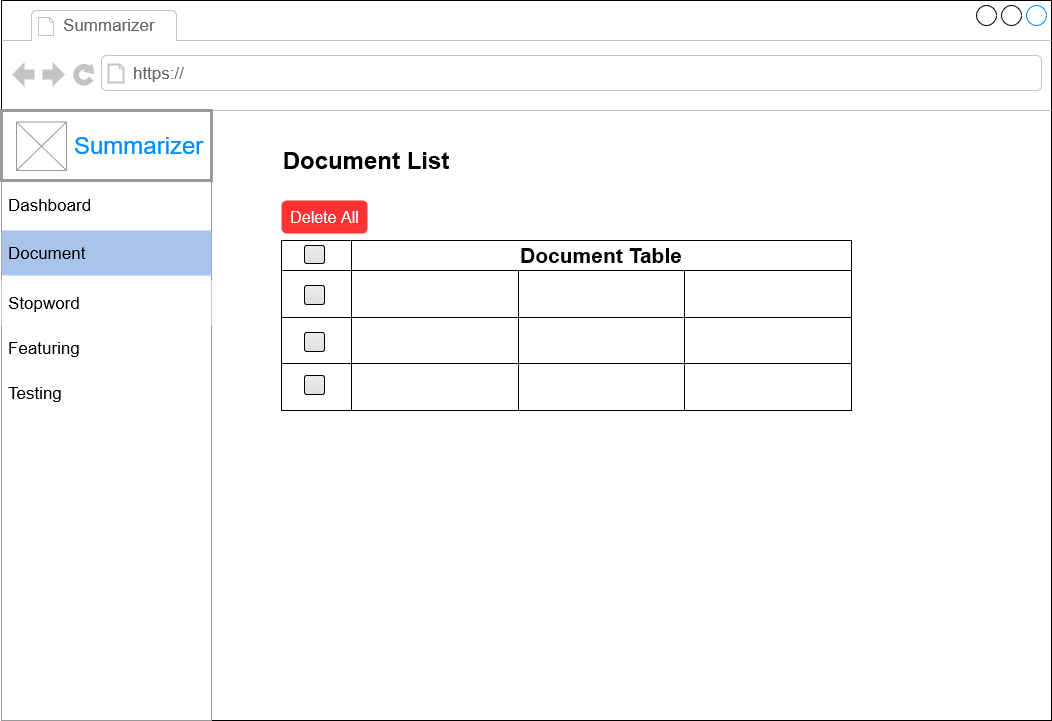
Pada alur proses diatas, dijelaskan langkah-langkah dalam pengimplementasian metode LSA untuk peringkasan teks pada dokumen.



### Perancangan Antar muka (Interface)

Mengolah dokumen

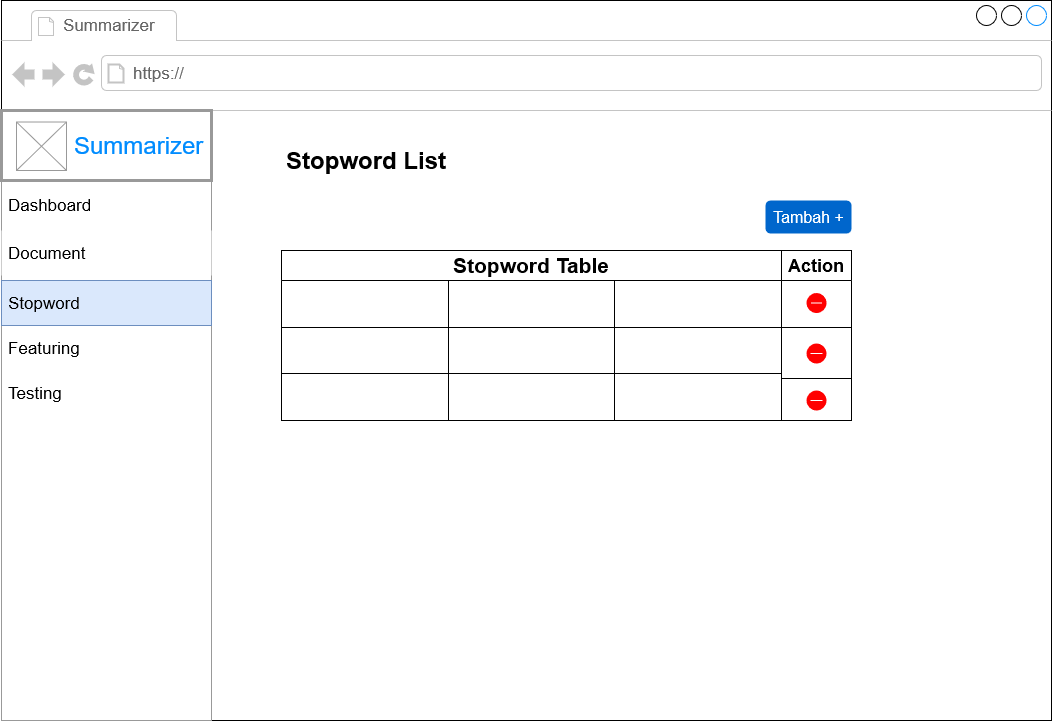
Pada halaman mengolah data dokumen, admin dan pegawai dapat melakukan kelola data seperti melihat dan hapus data. Untuk menghapus data *delete* yang terdapat pada kolom *action* di tabel. Tampilan antarmuka halaman ini dapat dilihat pada Gambar 4.11



Gambar 4.11 Mockup Mengolah Dokumen

Mengolah *Stopword*

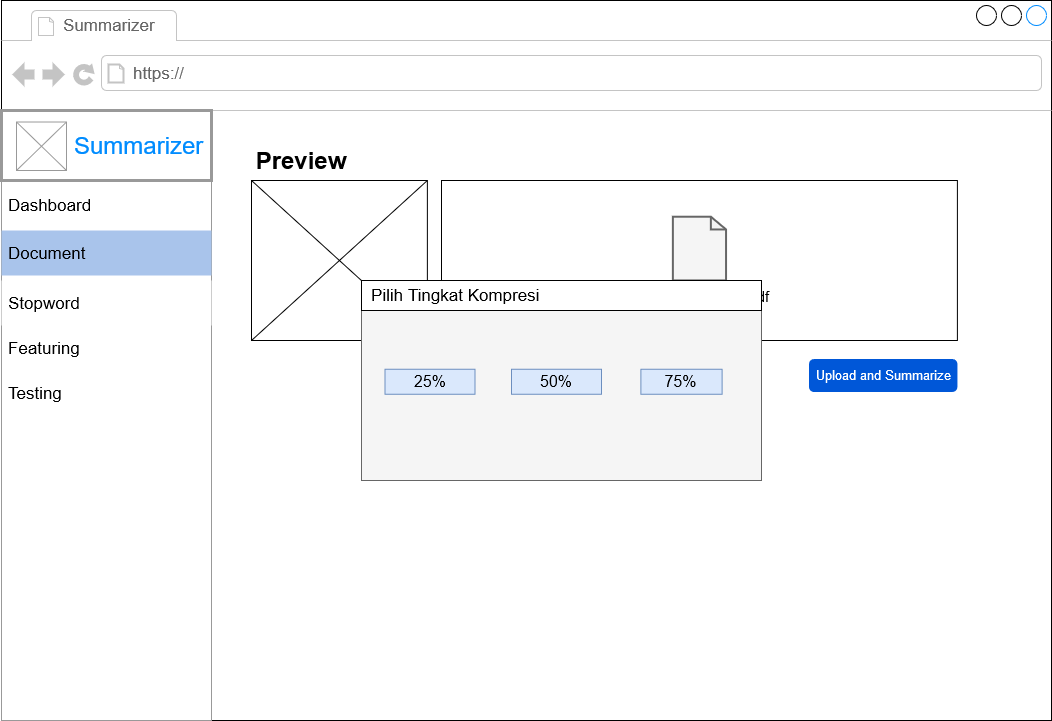
*Stopword* adalah kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna. Mengolah *stopword* merupakan proses melihat, menambah, dan menghapus data *stopword* yang terdapat pada *database.* Tampilan antarmuka halaman ini dapat dilihat pada Gambar 4.12



Gambar 4.12 Mockup Mengolah Stopword

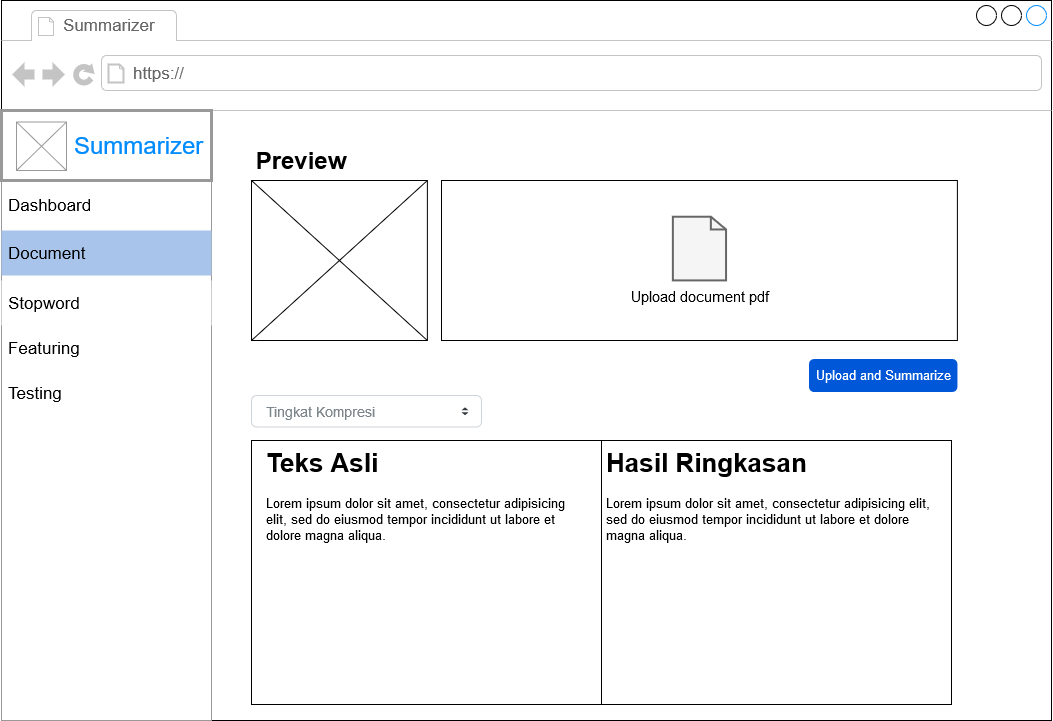
Melakukan Peringkasan

Untuk melakukan peringkasan dokumen, admin dan pegawai dapat mengunggah dokumen nota pembelaan yang akan diringkas. Selanjutnya, preview akan muncul, untuk memastikan data yang akan diunggah sudah benar. Untuk memproses maka admin dan pegawai dapat menekan tombol *upload and summarize*. Setelah mengunggah, maka admin dan pegawai dapat memilih tingkat kompresi panjangnya hasil ringkasan yang dinginkan. Tampilan antarmuka halaman ini dapat dilihat pada Gambar 4.13



Gambar 4.13 Mockup Pilih Kompresi

Sistem akan melakukan peringkasan dokumen. Jika telah selesai, maka hasil dari peringkasan akan ditampilkan dibawah *form* *upload* dokumen. Tampilan antarmuka halaman ini dapat dilihat pada Gambar 4.14



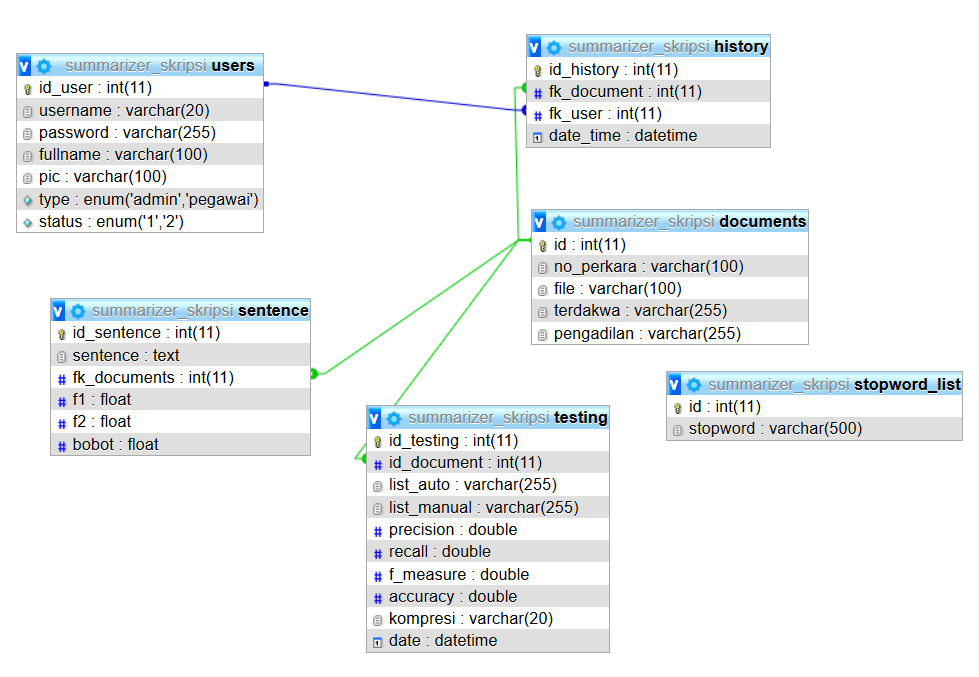
Gambar 4.14 Mockup Peringkasan Dokumen

# BAB V IMPLEMENTASI DAN PENGUJIAN



## Implementasi *Database*

*Database* merupakan salah satu komponen penting dalam sistem informasi, karena merupakan dasar dalam menyediakan informasi. *Database* sendiri adalah kumpulan dari item data yang saling berhubungan satu dengan yang lainnya yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu, yang kelak dapat dimanfaatkan kembali dengan cepat dan mudah (Minarni & Susanti, 2014). Pada proses pembangunan sistem, database diimplementasikan dengan nama summarizer\_skripsi yang memiliki 6 tabel diantaranya yaitu : *users, history, documents, sentence, testing* dan *stopword\_list*.



## Implementasi *Source Code*

Aplikasi ini dibuat menggunakan bahasa pemrograman php dan menggunakan *framework* *codeigniter.* Implementasi proses peringkasan berdasarkan pada analisis dan perancangan terdiri dari beberapa proses yaitu:

### Mengolah *Stopword*

Mengolah *stopword* merupakan proses melihat, menambah, dan menghapus data *stopword* yang terdapat pada *database*. Proses mengolah *stopword* dibagi menjadi beberapa potongan yaitu :

* + - 1. Melihat *stopword*

Proses ini menampilkan *stopword* yang diperoleh dari *database.*

|  |
| --- |
| **public** **function** index(){  $session\_data=$this->session->userdata('logged\_in');  $data['username']=$session\_data['username'];  $data['type']=$session\_data['type'];  $data['pic']=$session\_data['pic'];    $this->load->view('pages/stopword',$data);  }  **public** **function** getList(){  $data['data'] = $this->db->order\_by('id', 'asc')->get('stopword\_list')->result();    **echo** json\_encode($data);  } |

Source Code 5.1 Menampilkan Stopword

* + - 1. Menambah *stopword*

Proses ini digunakan untuk menambahkan data *stopword* kedalam database.

|  |
| --- |
| **public** **function** insertList(){  $array = **array**(  'stopword' => $this->input->post('stopword')  );  $data['data'] = $this->db->insert('stopword\_list', $array);    **echo** json\_encode($data);  } |

Source Code 5.2 Menambahkan Stopword

* + - 1. Mengubah *stopword*

Proses ini digunakan untuk mengubah data *stopword* kedalam *database*.

|  |
| --- |
| **public** **function** updateList(){  $where = **array**('id' => $this->input->post('id'));  $array = **array**(  'stopword' => $this->input->post('stopword')  );  $data['data'] = $this->db->where($where)->update('stopword\_list', $array);    **echo** json\_encode($data);  } |

Source Code 5.3 Mengubah Stopword

* + - 1. Menghapus *stopword*

Proses ini digunakan untuk menghapus data *stopword* kedalam *database*.

|  |
| --- |
| **public** **function** deleteList(){  $no = $this->input->post('id');  $data['data'] = $this->db->where('id', $no)->delete('stopword\_list');    **echo** json\_encode($data);  } |

Source Code 5.4 Menghapus Stopword

### Mengolah Pengguna

Mengolah pengguna merupakan proses melihat data, menambah data, mengubah status dan *reset* *password* pengguna yang terdapat pada *database*. Proses mengolah pengguna dibagi menjadi beberapa potongan yaitu :

* + - 1. Melihat data pengguna

Proses ini digunakan untuk melihat data pengguna yang tersimpandi *database*.

|  |
| --- |
| **public** **function** index(){  $session\_data=$this->session->userdata('logged\_in');  $data['username']=$session\_data['username'];  $data['type']=$session\_data['type'];  $data['pic']=$session\_data['pic'];  $this->load->view('pages/user',$data);  }  **public** **function** getList(){  $data['data'] = $this->db->order\_by('id\_user', 'asc')->get('users')->result();    **echo** json\_encode($data);  } |

Source Code 5.5 Melihat Data Pengguna

* + - 1. Menambah data pengguna

Proses ini digunakan untuk menambah data pengguna kedalam *database*.

|  |
| --- |
| **public** **function** insertList(){  $array = **array**(  'username' => $this->input->post('username'),  'password' => md5($this->input->post('username')),  'fullname' => $this->input->post('fullname'),  'pic' => 'user-no-photo.png',  'type' => $this->input->post('type'),  'status' => 2  );  $data['data'] = $this->db->insert('users', $array);  **echo** json\_encode($data);  } |

Source Code 5.6 Menambah data pengguna

* + - 1. Mengubah status pengguna

Proses ini digunakan untuk mengubah status pengguna yang tersimpan di *database*. Admin dapat memberikan atau mematikan hak akses pengguna dengan mengubah status tersebut.

|  |
| --- |
| **public** **function** updateStatus(){  $id = $this->input->post('id');  $result = $this->db->get\_where('users',**array**('id\_user' => $id))->result();  **if**($result[0]->status == 1){  $data = **array**(  'status' => 2  );  }**else**{  $data = **array**(  'status' => 1  );  }  $this->db->where('id\_user', $id);  $this->db->update('users', $data);  **echo** json\_encode($data);  } |

Source Code 5.7 Mengubah status pengguna

* + - 1. *Reset password* pengguna

Proses ini digunakan untuk mengatur ulang *password* pengguna yang telah terdaftar di *database*. *Password* *default* untuk pengguna sama dengan username yang digunakan.

|  |
| --- |
| **public** **function** resetPass(){  $id = $this->input->post('id');  $username = $this->input->post('username');  $data = **array**(  'password' => md5($username)  );  $this->db->where('id\_user', $id)->update('users',$data);  **echo** json\_encode($data);  } |

Source Code 5.8Me-reset password pengguna

### Mengolah Dokumen

Mengolah terdiri dari menampilkan dan menghapus dokumen.

* + - 1. Melihat dokumen

Proses ini digunakan untuk melihat dokumen yang tersimpandi *database*.

|  |
| --- |
| **public** **function** doc()  {  $session\_data=$this->session->userdata('logged\_in');  $data['username']=$session\_data['username'];  $data['type']=$session\_data['type'];  $data['pic']=$session\_data['pic'];  $this->load->view('pages/documents,$data);  }  **public** **function** getDocuments(){  $data['data'] = $this->db->order\_by('id', 'desc')->get('documents')->result();    **echo** json\_encode($data);  } |

Source Code 5.9 Melihat dokumen

* + - 1. Menghapus dokumen

Proses ini digunakan untuk menghapus dokumen yang tersimpandi *database*.

|  |
| --- |
| **public** **function** deleteDocuments(){  $no = $this->input->post('ids');  $data['data'] = $this->db->where\_in('id', $no)->delete('documents');    **echo** json\_encode($data);  } |

Source Code 5.10 Menghapus dokumen

### Melakukan Peringkasan Dokumen

Melakukan peringkasan merupakan tahapan untuk mengolah data dokumen agar menjadi ringkasan. Proses ini terdiri beberapa tahap yaitu;

* + - 1. Mengunggah dokumen

Proses ini digunakan untuk mengunggah dokumen yang tersimpandi *database*.

|  |
| --- |
| **public** **function** upload(){  $session\_data=$this->session->userdata('logged\_in');  $id\_user = $session\_data['id'];  $this->load->helper(**array**('form','url'));  $config['upload\_path'] = './assets/uploads/originalDocument';  $config['allowed\_types'] = 'pdf';  $config['max\_size'] = 0;  $this->load->library('upload', $config);  **if** (!$this->upload->do\_upload('file')){  $data['error'] = $this->upload->display\_errors();  $this->load->view('pages/upload', $data);  }**else**{  $upload\_data = $this->upload->data();  $file\_name = $upload\_data['file\_name'];  $data\_doc = **array**(  'file' => $file\_name  );    $insert = $this->db->insert('documents',$data\_doc);  $id\_doc = $this->db->insert\_id();  $data\_history = **array**(  'fk\_document' => $id\_doc,  'fk\_user' => $id\_user  );  $insert = $this->db->insert('history', $data\_history);  $kompresi = $this->input->post('kompresi');  $this->pdf($file\_name, $id\_doc);  **echo** json\_encode($id\_doc);  }  } |

Source Code 5.11 Mengunggah dokumen

Proses dibawah ini digunakan mengekstraksi teks yang berformat pdf menjadi teks yang dapat diolah. Pada pengekstrakan teks, menggunakan *library* *PDFParser*.

|  |
| --- |
| $server\_file = base\_url('assets/uploads/originalDocument/'.$file\_name);  $parser = **new** \Smalot\PdfParser\Parser();  $pdf = $parser->parseFile($server\_file);  **if** ($pdf != "") {  $original\_text = $pdf->getText();  **if** ($original\_text != "") {  $text = nl2br($original\_text);  $text = $this->clean\_ascii\_characters($text);  $check\_text = preg\_split('/\s+/', $text, -1, PREG\_SPLIT\_NO\_EMPTY);  $no\_spacing\_error = 0;  $excessive\_spacing\_error = 0;  **foreach**($check\_text **as** $word\_key => $word) {  **if** (strlen($word) >= 30) {  $no\_spacing\_error++;  } **else** **if** (strlen($word) == 1) {  **if** (preg\_match('/^[A-Za-z]+$/', $word)) {  $excessive\_spacing\_error++;  }  }  }  } |

Source Code 5.12 Mengubah pdf menjadi teks yang dapat diolah.

Proses dibawah ini digunakan untuk memberi batasan teks yang akan diekstraksi atau diolah.

|  |
| --- |
| $beforeNo = 'NOMOR:';  $afterNo = 'PENGADILAN';  $noPerkara = substr($text, strpos($text, $beforeNo) + strlen($beforeNo));  $noPerkara = strstr($noPerkara, $afterNo, **true**);  $beforeTerdakwa = 'terdakwa';  $terdakwa = substr($text, strpos($text, $beforeTerdakwa) + strlen($beforeTerdakwa));  $hasil = implode(" ", array\_slice(explode(" ", $terdakwa), 1, 1));  $beforePengadilan = 'PENGADILAN';  $afterPengadilan = 'I. PENDAHULUAN';  $pengadilan = substr($text, strpos($text, $beforePengadilan) + strlen($beforePengadilan));  $pengadilan = strstr($pengadilan, $afterPengadilan, **true**);  $afterString = 'PENDAHULUAN';  $beforeString = 'II.';  $text = substr($text, strpos($text, $afterString) + strlen($afterString));  $text = strstr($text, $beforeString, **true**); |

Source Code 5.13 Batas Pengambilan teks yang diolah.

* + - 1. *Preprocessing*

Pada tahap *preprocessing* untuk proses *stemming* menggunakan *library* sastrawi.

|  |
| --- |
| **public** **function** process($text, $casefolding = **true**, $filtering = **true**, $stemming = **true**, $tokenizing = **true**, $stopword = **true**){  $stopwords\_removal = **array**();  **foreach** ($this->db->select('stopword')->get('stopword\_list')->result() **as** $key => $value) {  $stopwords\_removal[$key] = $value->stopword;  }  $GLOBALS['stopwords'] = $stopwords\_removal;  **if** ($casefolding) {  $text = strtolower($text);  }  **if** ($filtering) {  $text = preg\_replace("/[^a-zA-Z0-9\s- .]/", "", $text);  }  $stemmerFactory = **new** \Sastrawi\Stemmer\StemmerFactory();  $stemmer = $stemmerFactory->createStemmer();  **if** ($casefolding && $stemming) {  $text = $stemmer->stem($text);  }  **if** ($tokenizing) {  $text = explode(" ", $text);  }  **if** ($stopword) {  $text = array\_filter($text, **function** ($key) {  **return** !in\_array($key, $GLOBALS['stopwords']);  });  }  **return** $text;  } |

Source Code 5.14 Preprocessing

* + - 1. *TF-IDF*

Proses *TF-IDF* berdasarkan data yang didapatkan dalam proses *preprocessing*.

|  |
| --- |
| **function** tf\_idf($text = **null**, $id\_doc = **null**){  $text\_list\_word = [];  **foreach** ($text[0]['preprocessing\_sentences'] **as** $key => $value) {  $text\_list\_word = array\_merge($text\_list\_word, $value);  }  $text\_list\_word = array\_unique($text\_list\_word);  $matrix\_tf = [];  **foreach** ($text\_list\_word **as** $key\_word => $word) {  **foreach** ($text[0]['preprocessing\_sentences'] **as** $key\_text => $value\_text) {  $count\_word = 0;  **foreach** ($value\_text **as** $k => $v) {  **if** ($word == $v) {  $count\_word++;  }  }  $matrix\_tf[$key\_word][$key\_text] = $count\_word;  }  }  $text\_df = [];  **foreach** ($matrix\_tf **as** $key => $value) {  $df = 0;  **foreach** ($value **as** $k => $v) {  **if** ($v > 0) {  $df++;  }  }  $text\_df[$key] = $df;  }  $text\_dperdf = [];  $text\_idf = [];  $text\_idfplus1 = [];  $word\_count = count($text[0]['preprocessing\_sentences']);  **foreach** ($text\_df **as** $key => $value) {  $dperdf = $word\_count / $value;  $text\_dperdf[$key] = $dperdf;  $idf = log10($dperdf);  $text\_idf[$key] = $idf;  $text\_idfplus1[$key] = $idf + 1;  }  $matrix\_tfidf = [];  **foreach** ($matrix\_tf **as** $key => $value) {  **foreach** ($value **as** $k => $v) {  $matrix\_tfidf[$key][$k] = $v \* $text\_idfplus1[$key];  }  }  $array\_martix[] = [  'matrix\_tf' => $matrix\_tf,  'text\_idfplus1' => $text\_idfplus1,  'text\_df' => $text\_df,  'text\_idf' => $text\_idf,  'matrix\_tfidf' => $matrix\_tfidf,  'text\_list\_word' => $text\_list\_word  ];  **return** $array\_martix;  } |

Source Code 5.15 TF-IDF

* + - 1. Menghitung nilai kesamaan kata antar kalimat menggunakan metode *latent semantic analysis*.

Menghitung fitur kesamaan kata antar kalimat menggunakan metode *Cross* *method* *Latent* *Semantic* *Analysis*.

|  |
| --- |
| **function** lsa($matrix\_tfidf = **null**, $id\_sentences = **null**, $id\_doc = **null**){  $this->load->library("singularvaluedecomposition");  $svd = $this->singularvaluedecomposition->SVD($matrix\_tfidf);    $matrix\_Vt = $this->singularvaluedecomposition->matrixRound($svd['V']);  $matrix\_S = $this->singularvaluedecomposition->matrixRound($svd['S']);  $rows\_matrix\_Vt = count($matrix\_Vt);  $cols\_matrix\_Vt = count($matrix\_Vt[0]);    $rows\_matrix\_S = count($matrix\_S);  $cols\_matrix\_S = count($matrix\_S[0]);    **if**($cols\_matrix\_Vt == $rows\_matrix\_S){  **for**($i = 0; $i < $rows\_matrix\_Vt; $i++){  **for**($j = 0; $j < $cols\_matrix\_S; $j++){  **for**($p = 0; $p < $cols\_matrix\_Vt; $p++){  $matrix\_LSA[$i][$j] += $matrix\_Vt[$i][$p] \* $matrix\_S[$p][$j];  }  }  }  }  $lsa\_Length = [];  **foreach** ($matrix\_LSA **as** $key => $value) {  **foreach** ($value **as** $k => $v) {  $lsa\_Length[$k] += $v;  }  }  **foreach** ($lsa\_Length **as** $key => $value) {  **if**($value >= 0){  $value = sqrt($value);  }**else**{  $value = -sqrt(abs($value));  }  $data = **array**(  'f1' => $value  );  $this->db->where('id\_sentence', $id\_sentences[$key]);  $this->db->update('sentence', $data);  }  **return** $this->db->where('fk\_documents', $id\_doc)->get('sentence')->result();  } |

Source Code 5.16 Metode LSA

### Melakukan Pengujian

Melakukan pengujian dilakukan dengan cara membandingkan dokumen hasil ringkasan otomatis dengan hasil ringkasan manual.

* + - 1. Menghitung *precision*

*Precision* adalah berapa banyak dokumen yang berhasil diambil oleh sistem. Dibawah ini merupakan proses perhitungan *precision*.

|  |
| --- |
| **public** **function** precision($summaryAuto = **null**, $summaryManual = **null**){  $array\_join = array\_intersect($summaryAuto, $summaryManual);  $tpfp = count($summaryAuto);  $tp = count($array\_join);    **return** $precision = $tp / $tpfp;  } |

Source Code 5.17 Menghitung precision

* + - 1. Menghitung *recall*

*Recall* ialah kemampuan untuk mengambil peringkat teratas yang sebagian besar relevan (benar). Dibawah ini merupakan proses perhitungan *recall*.

|  |
| --- |
| **public** **function** recall($summaryAuto = **null**, $summaryManual = **null**){  $array\_join = array\_intersect($summaryAuto, $summaryManual);  $tpfn = count($summaryManual);  $tp = count($array\_join);  **return** $recall = $tp / $tpfn;  } |

Source Code 5.18 Menghitung Recall

* + - 1. Menghitung *f-measure*

*F-measure* digunakan untuk mengukur kualitas *recall* dan *precision*. Dibawah ini merupakan proses perhitungan *f-measure*.

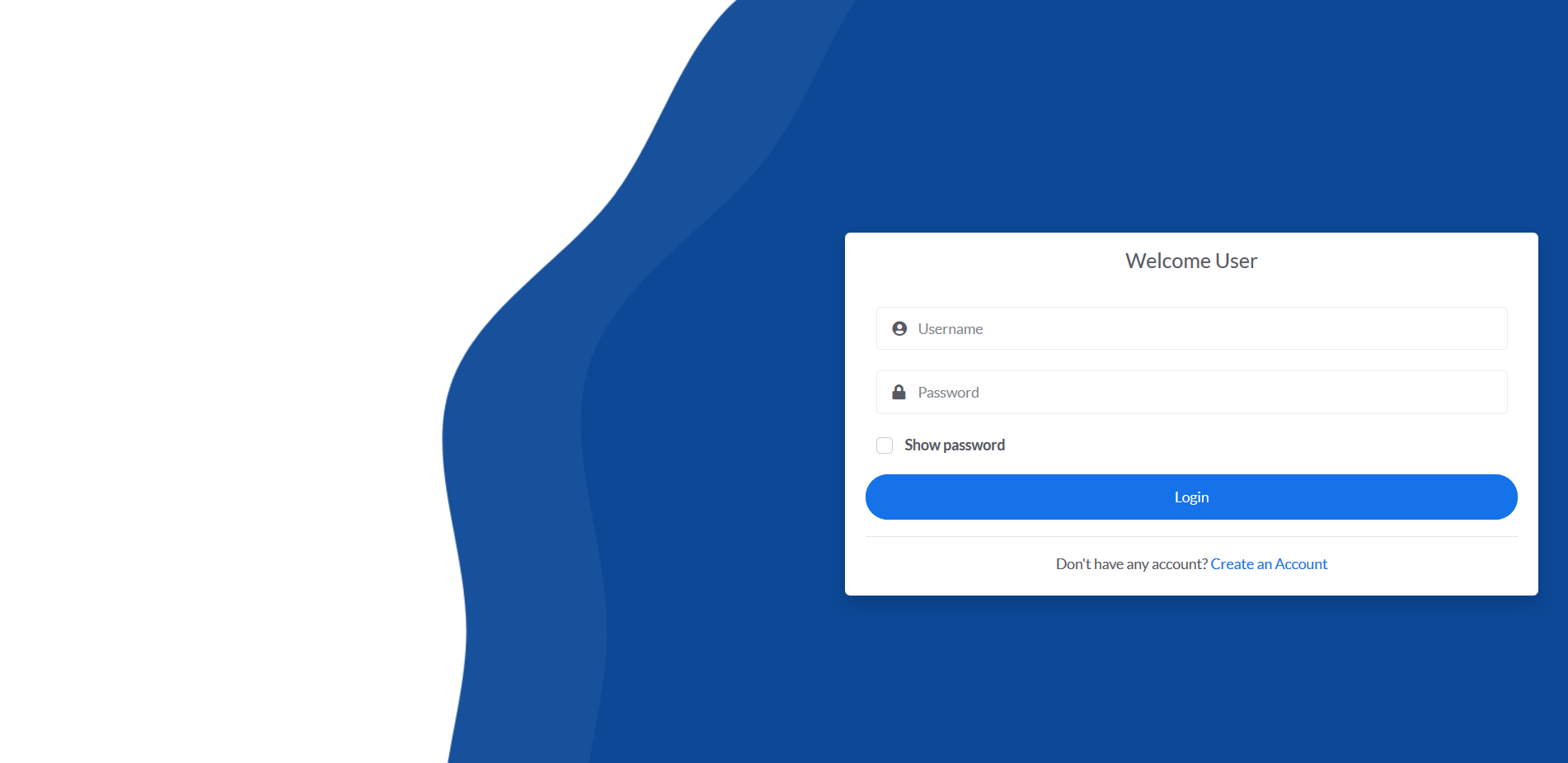
|  |
| --- |
| **public** **function** f\_measure($precision = **null**, $recall = **null**){  **return** $f\_measure = 2\* ($precision\*$recall) / ($precision+$recall);  } |

Source Code 5.19 Menghitung f-measure

## Implementasi Tampilan

### *Login*

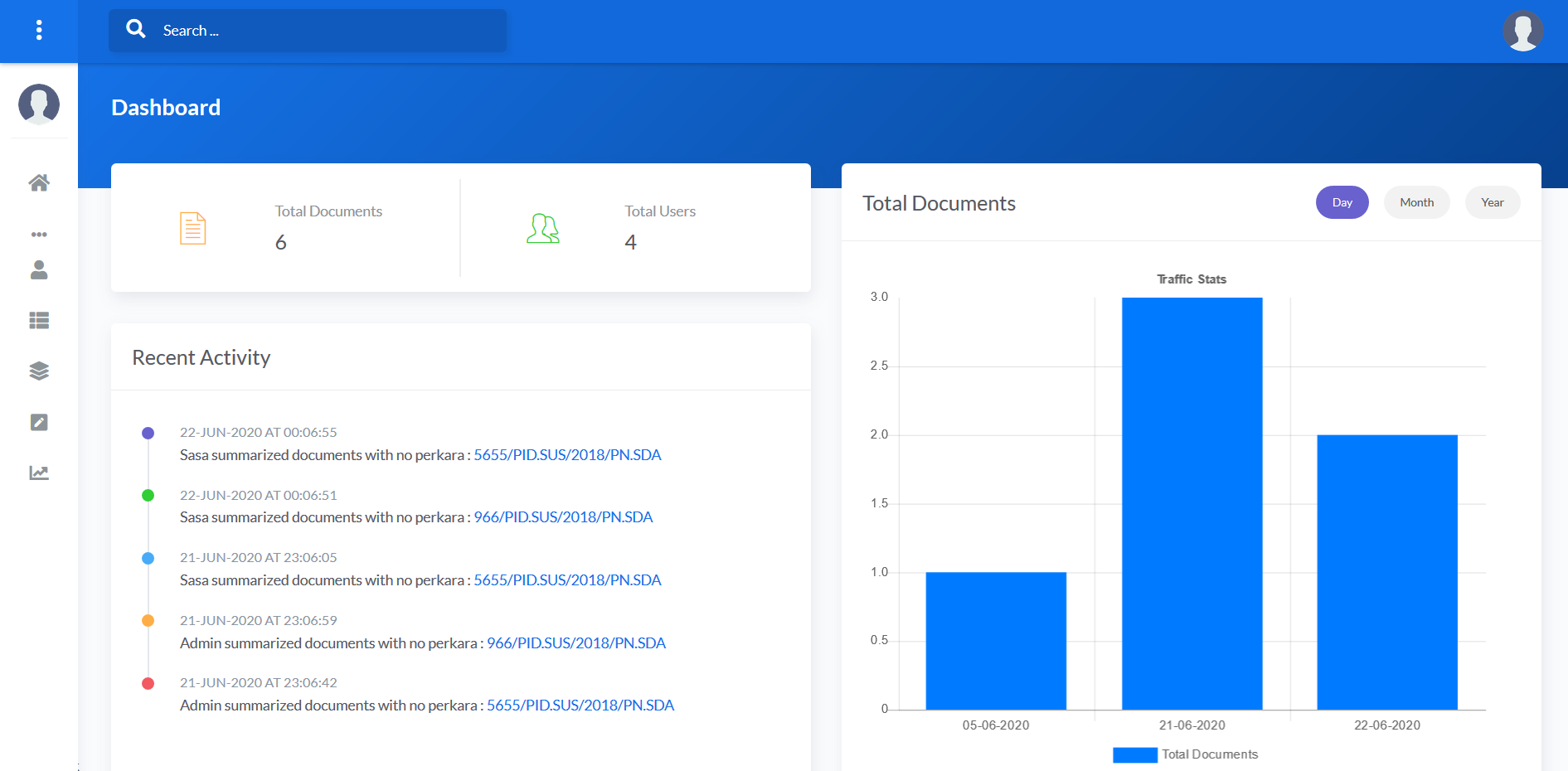
Tampilan *login* ini diperuntukan oleh admin dan pegawai untuk dapat mengakses *website* dalam melakukan peringkasan dokumen secara otomatis. Tampilan antarmuka halaman *login* dapat dilihat pada Gambar 5.1



Gambar 5.1 Tampilan Login

### *Dashboard*

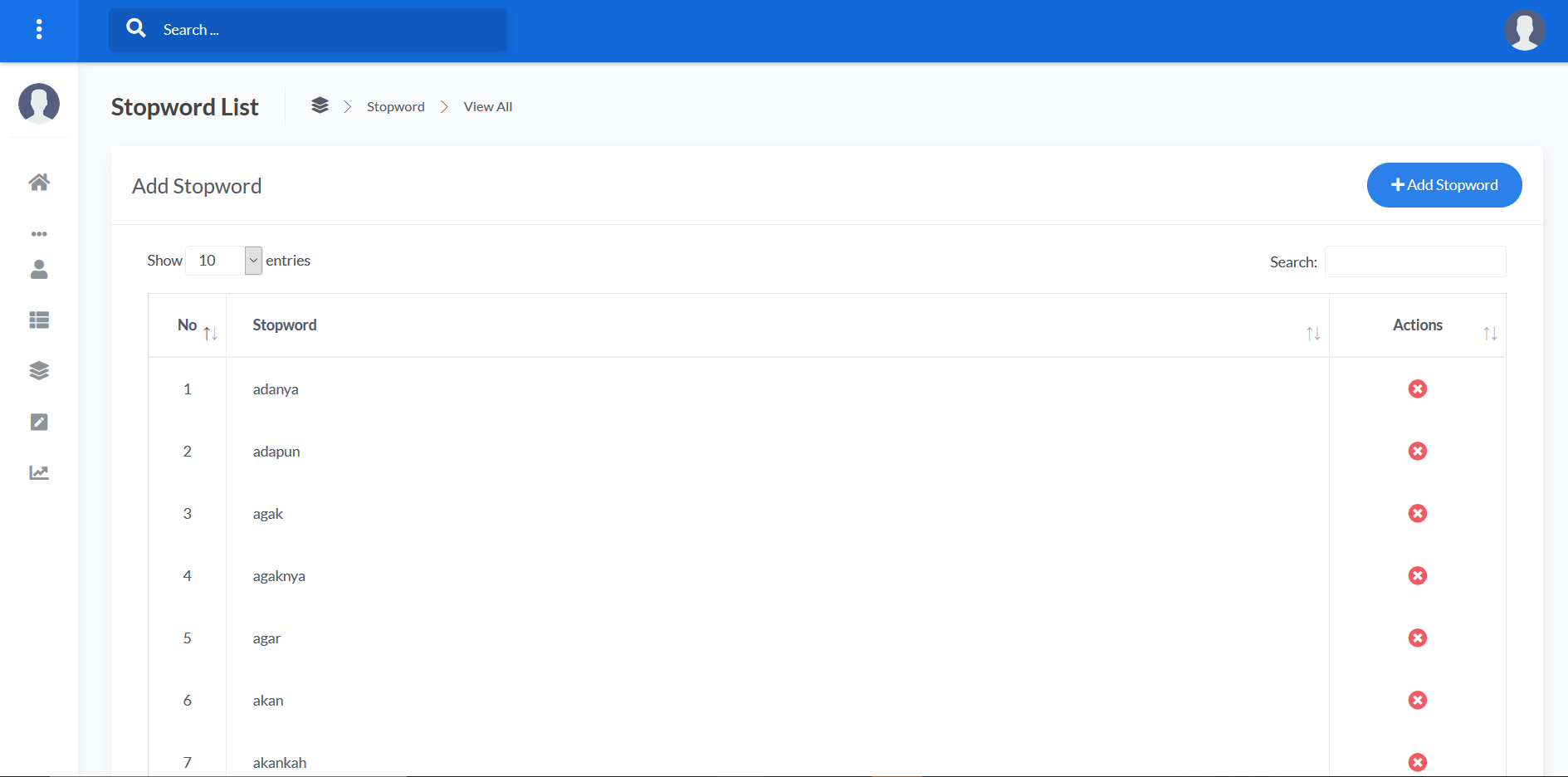
Tampilan pertama yang ditampilkan oleh sistem setelah admin melakukan login atau pengguna saat mengakses *website* sistem ini adalah halaman dashboard. Tampilan antarmuka halaman dashboard dapat dilihat pada Gambar 5.2



Gambar 5.2 Tampilan Dashboard

### Mengolah *stopword*

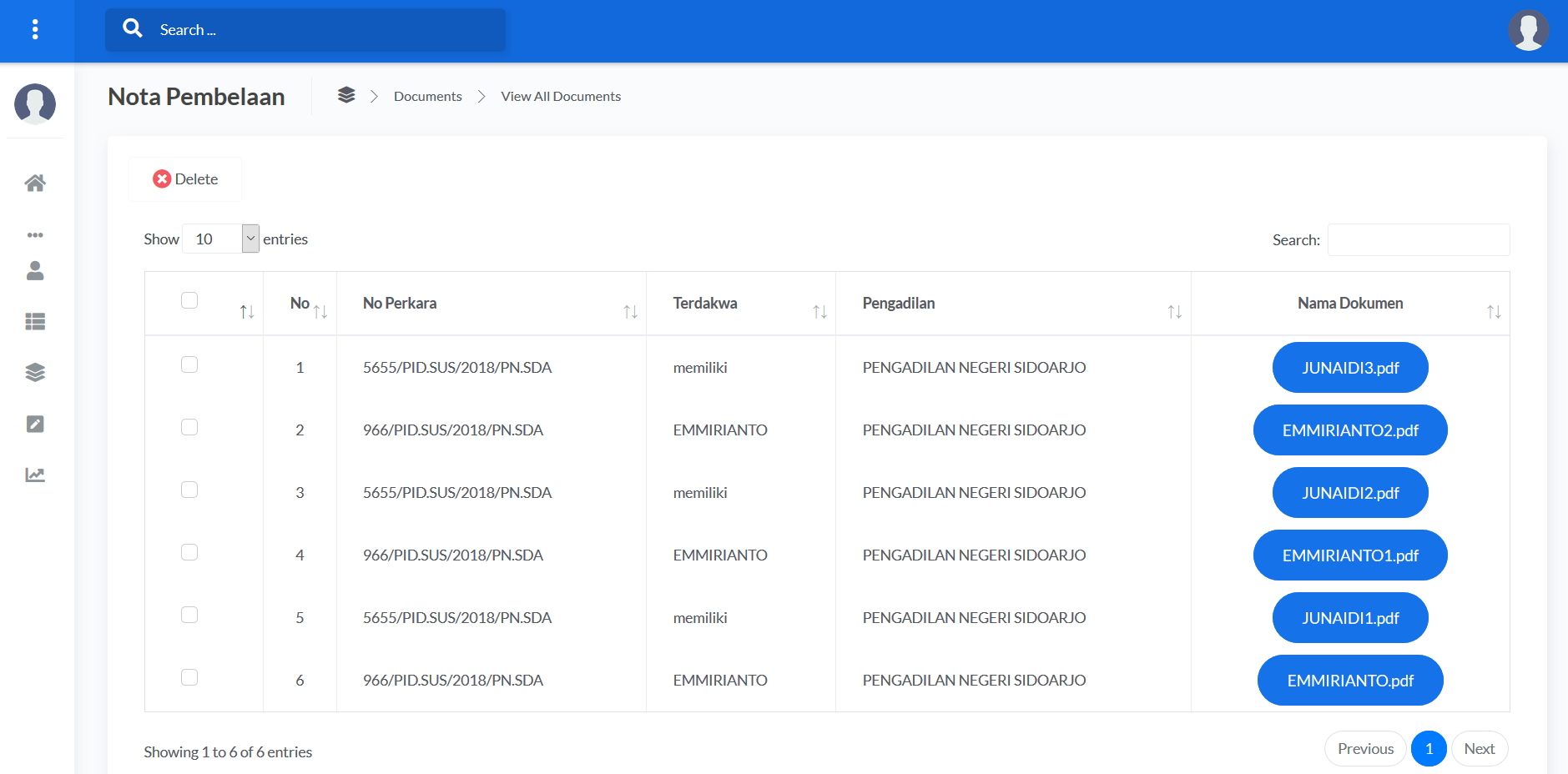
Pada halaman mengolah data *stopword*, admin dapat melakukan kelola data berupa tambah dan hapus data. Untuk menambahkan data dapat dilakukan dengan menekan tombol add data. Untuk menghapus data *delete* yang terdapat pada kolom *action* di tabel. Tampilan antarmuka halaman ini dapat dilihat pada Gambar 5.3



Gambar 5.3 Tampilan Mengolah Stopword

### Mengolah dokumen

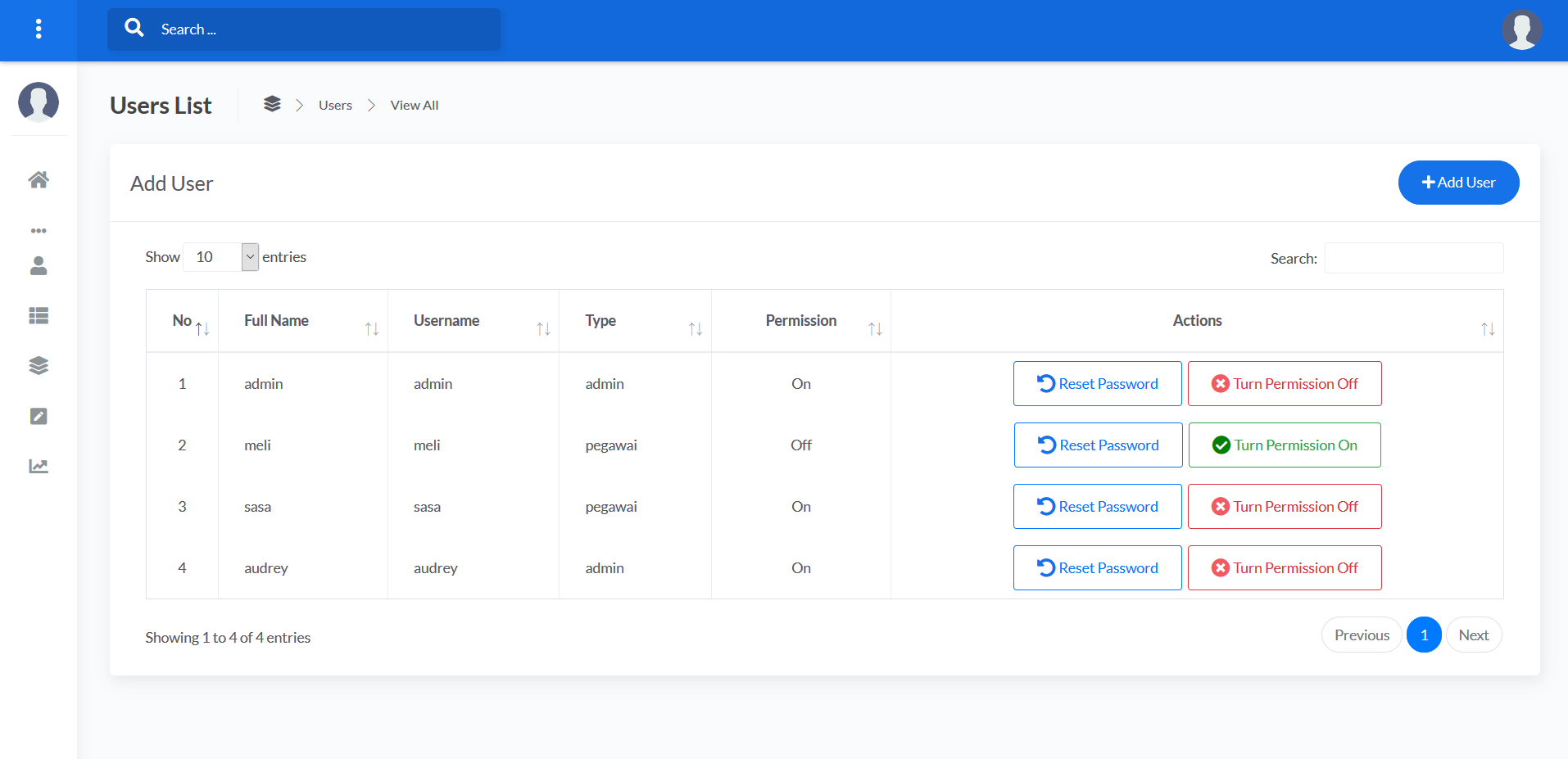
Pada halaman mengolah data dokumen, admin dan pegawai dapat melakukan kelola data seperti melihat dan hapus data. Untuk menghapus data *delete* yang terdapat pada kolom *action* di tabel. Tampilan antarmuka halaman ini dapat dilihat pada Gambar 5.4



Gambar 5.4 Tampilan Mengolah Dokumen

### Mengolah pengguna

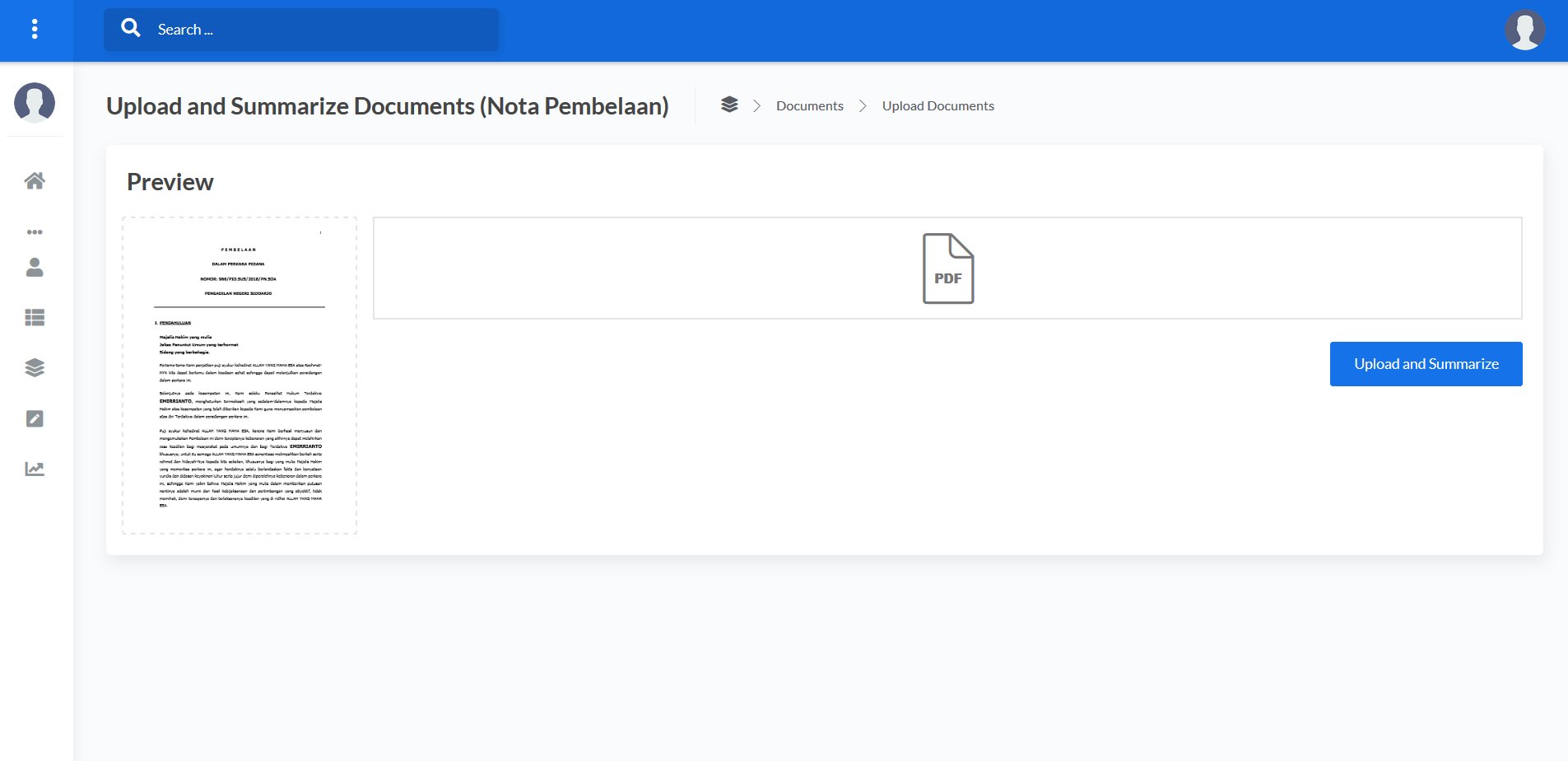
Pada halaman mengolah data pengguna, admin dapat melakukan kelola data berupa tambah, mengatur ulang password serta mematikan atau menyalakan hak akses pengguna. Untuk menambahkan data dapat dilakukan dengan menekan tombol *add user*. Untuk mengatur ulang *password* ke *default* dapat dilakukan dengan menekan tombol *reset* *password* pada kolom *action*. Sedangkan untuk mematikan atau menyalakan hak akses pengguna dapat dilakukan dengan menekan tombol *turn permission on* atau *turn permission off* yang terdapat pada kolom *action* di tabel. Tampilan antarmuka halaman ini dapat dilihat pada Gambar 5.5



Gambar 5.5 Tampilan Mengolah Pengguna

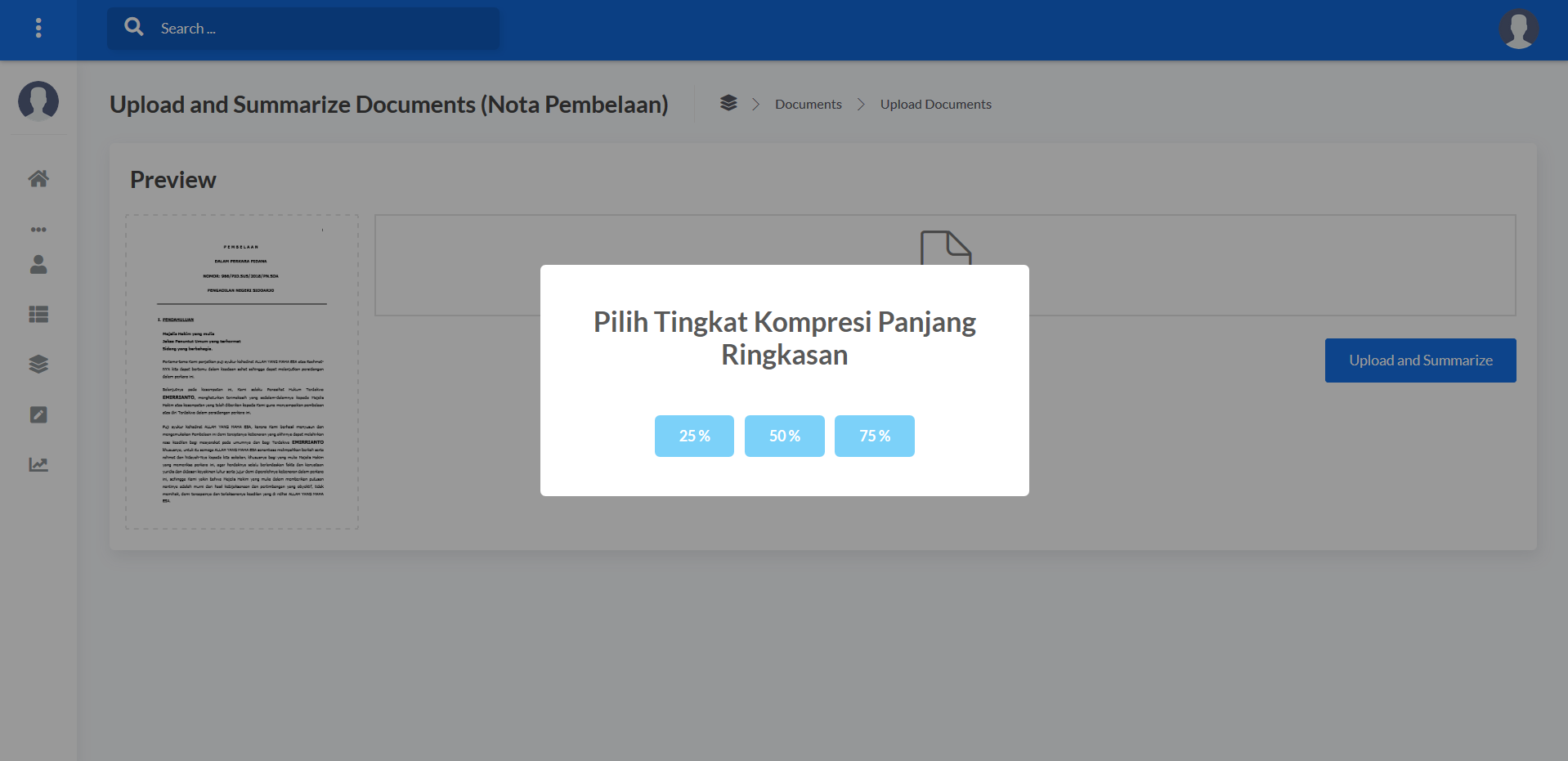
### Melakukan peringkasan

Untuk melakukan peringkasan dokumen, admin dan pegawai dapat mengunggah dokumen nota pembelaan yang akan diringkas. Selanjutnya, preview akan muncul, untuk memastikan data yang akan diunggah sudah benar. Untuk memproses maka admin dan pegawai dapat menekan tombol *upload and summarize*. Tampilan antarmuka halaman ini dapat dilihat pada Gambar 5.6



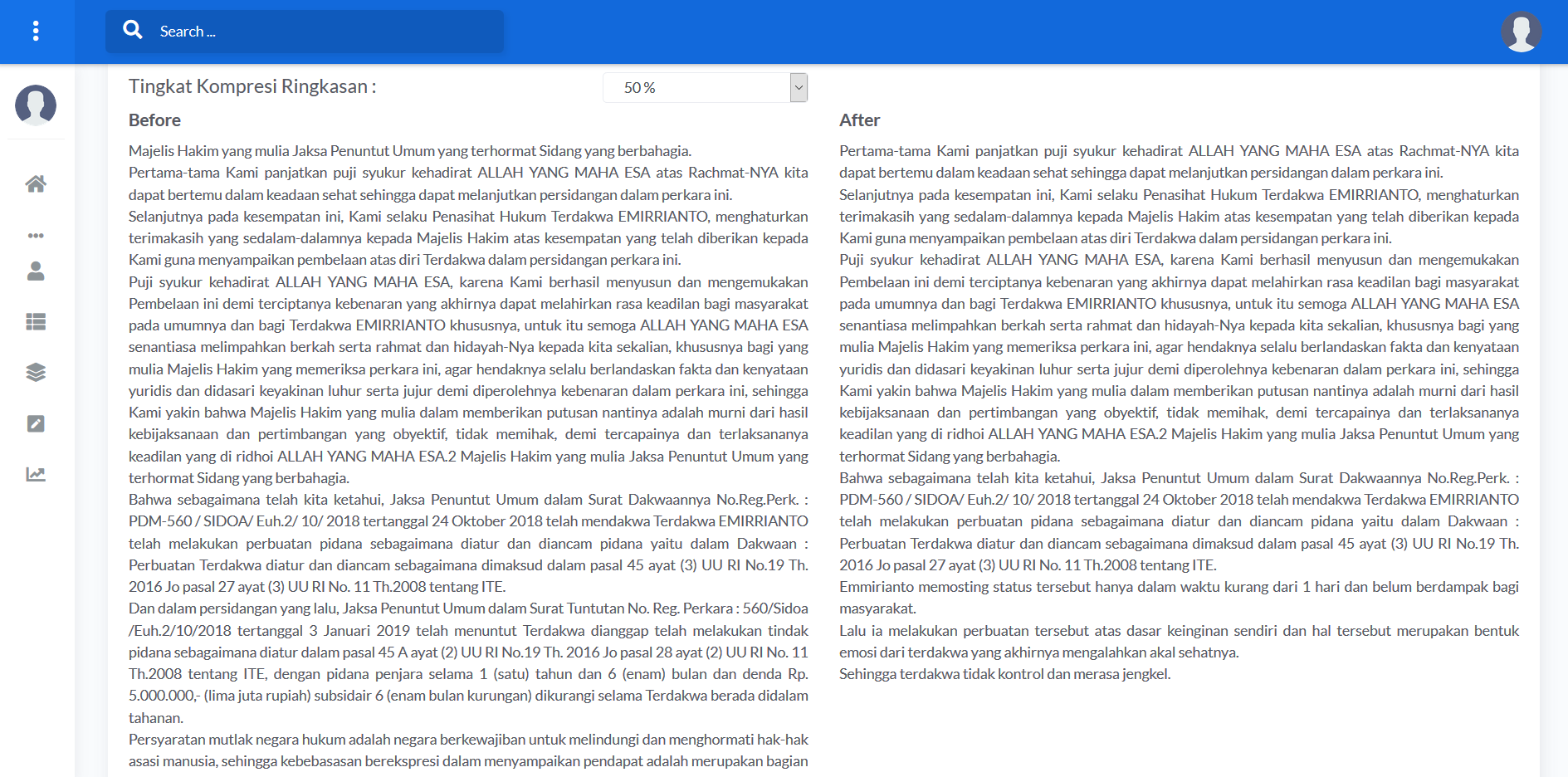
Gambar 5.6 Tampilan Pengunggahan Dokumen

Setelah mengunggah, maka admin dan pegawai dapat memilih tingkat kompresi panjangnya hasil ringkasan yang dinginkan. Tampilan antarmuka halaman ini dapat dilihat pada Gambar 5.7



Gambar 5.7 Tampilan Pilih Tingkat Kompresi

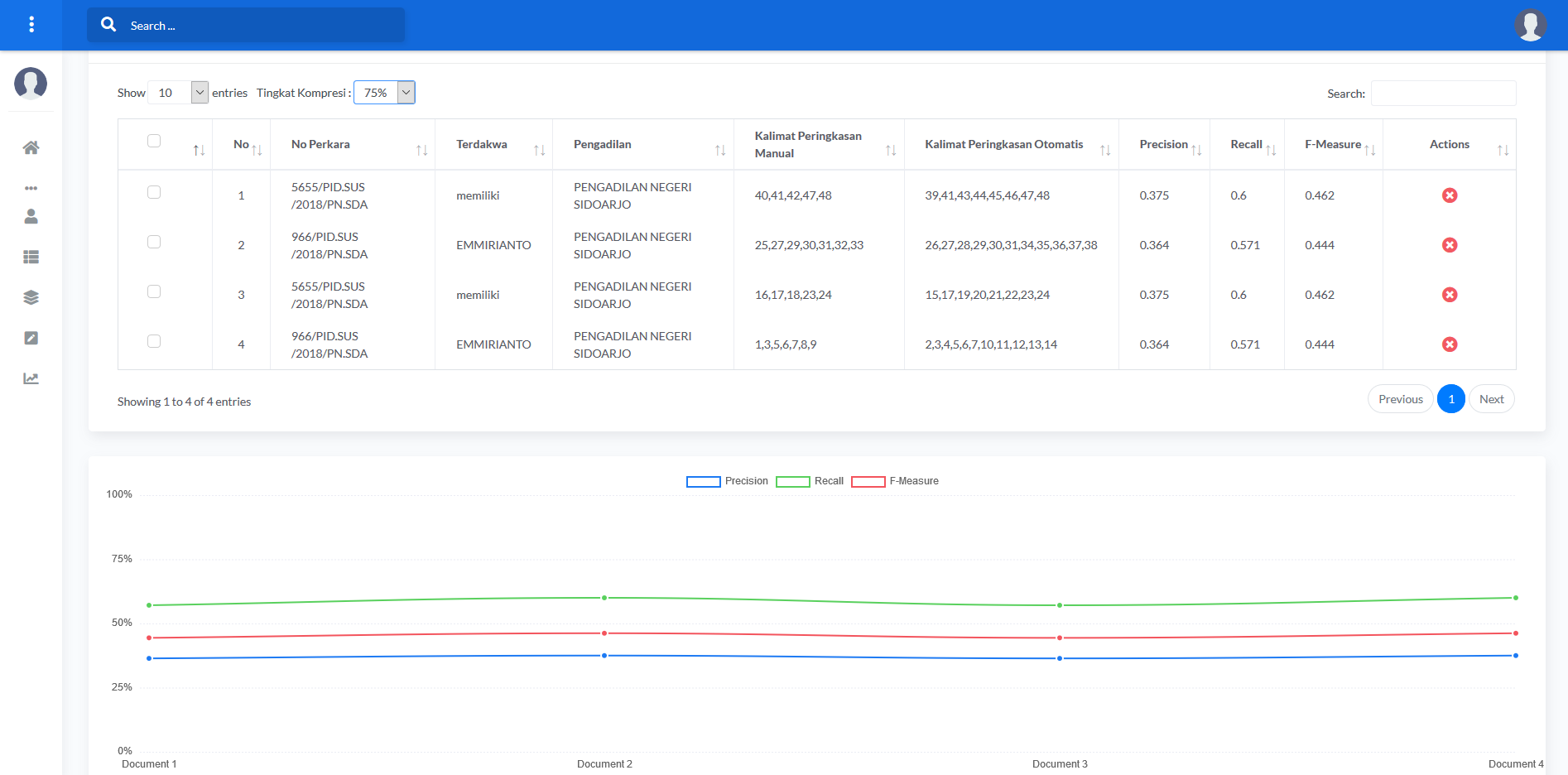
Sistem akan melakukan peringkasan dokumen. Jika telah selesai, maka hasil dari peringkasan akan ditampilkan dibawah *form* *upload* dokumen. Tampilan antarmuka halaman ini dapat dilihat pada Gambar 5.8



Gambar 5.8 Hasil Peringkasan Dokumen

### Melakukan pengujian

Halaman pengujian ini difungsikan untuk menampilkan percobaan yang dilakukan pada pengujian metode sistem ini yang dapat dilihat oleh admin. Pada halaman ini ditampilkan *line chart* dokumen apa saja yang telah diuji serta tabel yang diletakkan diatas grafik. Tampilan antarmuka halaman tersebut dapat dilihat pada Gambar 5.9



Gambar 5.9 Tampilan Pengujian

## Pengujian Fungsinalitas Sistem

Pengujian fungsionalitas sistem dilakukan dengan cara menjalankan tiap fitur dalam aplikasi dan melihat kesesuaian hasil yang terjadi dengan hasil yang diharapkan

Tabel 5.1 Tabel pengujian fungsionalitas sistem

|  |  |  |  |
| --- | --- | --- | --- |
| No | *Use Case* | Keterangan | Status |
| 1. | Mengolah *stopword* | Admin dapat melihat, tambah, edit, hapus pada data bobot. | Sesuai |
| 2. | Mengolah pengguna | Admin dapat melihat, tambah, edit, hapus pada data bobot. | Sesuai |
| 3. | Mengolah dokumen | Admin dan pengguna lain dapat melihat daftar dokumen dan menghapus dokumen. | Sesuai |
| 4. | Melakukan peringkasan | Admin dan pengguna lain dapat melakukan peringkasan dengan cara menghitung kesamaan kata antar kalimat menggunakan metode LSA*.* | Sesuai |
| 5. | Melakukan pengujian | Admin dapat melakukan pengujian dengan melakukan mengunggah peringkasan dokumen secara manual. Hasil perhitungan akurasi akan tampil setelah melakukan penggunggahan. | Sesuai |

## Pengujian Akurasi Sistem

Pengujian akurasi dilakukan dengan cara membandingkan peringkasan teks otomatis dengan peringkasan teks manual yang telah divalidasi oleh pakar. Pengujian yang dilakukan menggunakan pengujian *precision*, *recall*, dan *f-measure*. Terdapat beberapa skenario pengujian yang akan dilakukan diantaranya :

(Sesuai scenario pada bab 3)

# BAB VI HASIL DAN PEMBAHASAN

# BAB VII KESIMPULAN DAN SARAN

## 6.1. Kesimpulan (menjawab rumusan masalah, selanjtnya kesimpulan setelah pengejian poin poin dalam bentuk paragraf)

## 6.2 Saran (kendala)



# DAFTAR PUSTAKA (minimal 10)

# LAMPIRAN